DOCUMENT RESUME

ED 094 777 IR 000 978

AUTHOR Burns, Harris, Jr., Ed.; Esbin, John H., Ed.
TITLE Computing and Colleges. Proceedings of a SIGUCC Symposium, June 21-22, 1973, Claremont Colleges.
INSTITUTION Claremont Univ. Center, Calif.
PUB DATE Jun 73
NOTE 56p.; Some pages may reproduce poorly due to size of print and legibility of original document

EDRS PRICE MF-$0.75 HC-$3.15 PLUS POSTAGE
DESCRIPTORS *Colleges; Computer Oriented Programs; *Computer Programs; Computers; *Computer Science; *Computer Science Education; Conference Reports; *Higher Education; Symposia; Universities

ABSTRACT

At a symposium on computers and colleges, speakers addressed a variety of topics relating to the current status of computing, its effect on students, its academic implications, and its technical details. Small colleges and networks were the main foci for the first speakers. The role of amateurs and the variation of user services were also discussed. For the next three speakers the focus was on the future of computer science programs of California. Another three speakers discussed the academic future of computer science. The role of computers in teaching and learning, both now and in the future, was addressed by two participants. Finally, there was a discussion of computer software for college administrative needs and some thoughts on how to make do with what one has. (WH)

COMPUTING AND COLLEGES

Proceedings of a SIGUCC Symposium

June 21-22, 1973

Claremont Colleges

Edited By

Harris Burns, Jr.
Randolph-Macon College

John H. Esbin
The University of Iowa

COMPUTING AND COLLEGES

Proceedings of a SIGUCC Symposium

June 21-22, 1973

Claremont Colleges

Edited By

Harris Burns, Jr.
Randolph - Macon College

John H. Esbin
University of Iowa

TABLE OF CONTENTS

SECTION I

WHERE ARE WE NOW?

WHAT IS THE STATUS OF THE SMALL COLLEGE COMPUTER CENTER

Sister Mary K. Keller
Clarke College
Dubuque, Iowa

When we set up the Clarke College computer center in 1965, at a small liberal arts college for women, we were considered ambitious, if not a little extravagant, by other small colleges of which Iowa has more than average. Now, in the intervening years the picture has changed. Practically all colleges large and small have acquired some computer facility. Several in Iowa, like ourselves, have an 1130 system, and others who lived within the range of the Iowa City network have a terminal connection to the University of Iowa's 360/65. Although I do not have a survey on every institution in Iowa, it is safe to say that most of the state has some access to a computer by 1973.

Our own equipment includes the 1130 system with a disk drive, card reader and punch, paper tape reader and a 600 line per minute printer. The 1130 serves as a terminal to the University of Iowa's computer center. We have, in addition, an IBM 2741 terminal which uses the same communication line for interactive computing when we are not operating in the remote batch mode. We own all the equipment except the 2741, four keypunches, and an ancient sorter. The decision to buy, in 1965, was a difficult one to make, but we have not regretted it. We would have paid out more than twice the cost over eight years and we still have a viable system capable of expansion.

How does this configuration meet our needs? We can state that without a doubt we have ample computing facility for our present design with sufficient flexibility open to us for future plans. Present design, which is also close to our original plan calls for a curriculum in the computer sciences which would provide professional training for students who wished to pursue computer-related careers either immediately after completion of their first degree, or after graduate work in the field, and secondly to provide facilities and programming support for all faculty and students for both research and instruction in any field.

I would like to point out immediately that, given the present state of the art and current practice in business and industry, the 1130, by itself, would not adequately support career education as described. However, the ample facilities located at the University of Iowa to which we have 24 hour a day access do provide for our professional needs at a cost that we can afford. Without the network, such facilities would not be possible.

The large facilities even have certain advantages attached to their remoteness. An increasing number of industries, government, and research installations are operating in this mode. For future programmers and analysts this environment is not unlike the one in which they find themselves on their first job. On the other hand small businesses are still divided between a limited in-house system or a small terminal to a large timesharing facility. We have something close to all these approaches and students have an opportunity to experience them. They can come to appreciate the advantages and limitations of each, and most importantly, learn to work in several variations of systems.

While we are addressing ourselves to the question of what kind of equipment will support what kind of education, I might remark immediately that there are vocational institutions which are well-equipped to provide a highly technical environment, much more so than liberal arts colleges such as ourselves. Why don't we leave it to them? My answer to that is -- we do. They train one type of person, we train another kind. Their graduates may be programmers, technicians, possibly business analysts. Ours become system designers, managers, and possibly innovators in computer applications. And those that continue their education in the graduate schools have the preparation to do so. We believe the difference lies in a broader education.

At Clarke there is no computer science major, only a joint major with some other field. My experience with the progress of our graduates over eight years leads me to continue to recommend this approach to other liberal arts colleges, and most especially to those that are small. How are you different -- what is special about your center? I am continually challenged. Now the popular ad says that if you rank second customer-wise, you have to try harder; but when you rank 500th or worse trying won't do. You have to fill a special need. That special need, as I see it, is the integration of the computer sciences with other fields of knowledge in career preparation. The large universities could do this, but strangely enough, usually do not, and the technical, vocational institutions lack the capacity. That leaves us. At that point we can talk about trying harder. One way to do this is through a joint major where the integration of two fields is carefully planned by means of personal contact with the student, their future employers, field experience as part of curriculum, and cooperative planning between college departments. We do this and it is within the reach of other small colleges. In fact, smallness facilitates some of this integration and helps to explain its absence in larger universities.

The second function of a computer center, and for some small colleges the only one, is to provide an educational tool which is important both in methods of instruction and in the experience of a liberally educated person. The presence of something called a computer center with some dedicated students mingling with the rest, in itself contributes to this function, but does not support it entirely. Elaboration on the integration of computer usage in the curriculum would require a separate paper. I will confine myself to a few pertinent remarks.

When some limited computer facilities became available on a wider scale in the late 60's, for many teachers any sort of introduction to the computer by way of an exercise or unit seemed to be an enhancement of a course. Sophistication did develop though its course has been slow. Computer for computer's sake in any course has given way to many modes of instruction which might be classified as "computer-extended". A question could be raised as to the extent of its penetration into curricula, its real impact and effectiveness in American education in 1973. Perhaps we cannot know for a generation, and possibly the effects cannot be isolated.

One of the problems which still limits effective and widespread use of the computer as an educational tool is obviously cost. For every Dartmouth with its proliferation of terminals (and expertise) there are hundreds of Clearblue U's lucky to have one terminal and one enthusiastic instructor. Hopefully this picture will change. In the meantime, with present resources all is not lost.

My strong recommendation given previously on the education of computer professionals in a liberal arts environment is possible with resources such as described for Clarke College. Even less in equipment will enhance the education of students in other fields. To be specific, any computer facility, however limited, can make instruction in a variety of fields more research-oriented. Even though an adequate number of interactive terminals may not be available to deliver the impact of instant reenforcement, batch mode is not all that bad. In certain cases it has some things to recommend it. First of all, turnaround in a small center may be less than a half hour. Our standard is one class period. That is, the student may leave a deck at the beginning of a class period and pick up the results 50 minutes later. Not instant, but certainly better feed-back than most other modes of instruction supply. Moreover, the cheapest storage for small personal research files is a deck of cards. This is true at least in our environment. The card deck is a simple and easily understood introduction to a data file. There is a clear distinction between data and program. Students quickly graduate to disk and magnetic tape files from this beginning with a better appreciation of the advantages of magnetic storage.

In the batch mode approach the computer is more than a keyboard to the beginner. It is a system of interesting components which they manipulate. Of course it is true that this concept is not necessary to the use of the computer in most courses other than those in the computer sciences, but in the absence of multiple terminals it does supply some excitement, and is useful knowledge for citizens in general. At the same time the computer is still being used as a tool for research and problem solving in the general sense. The key to success here is not hardware but imaginative instructors.

In conclusion, it would appear that we are going forward in the use of computers in undergraduate instruction, but at a rather slow pace. The concept is accepted even in most small colleges with limited resources. The availability of expertise and hardware on the local level determine the amount of integration in course work and the kind of computer sciences program which can be supported. Wider availability of these components is on the horizon, but if past history is used as a predictor, the horizon will be approached very gradually.

REQUIREMENTS

For Professional Career or Graduate Study in Computer Sciences: 107, 131, 136, 156, 158, 160.

For Mathematics Students: 127, 155, 156, 158.

For Physical Science Students: 127, 156.

For Social Sciences: 136, 156.

For Mathematics Teachers: 109.

For Joint Major in Math Computer Sciences: See Mathematics Department.

AREA PROGRAM COURSES

Computer Sciences courses in Area V include 005, 127, 128, 136, 141.

CS 005 Introduction to Computer Science 3 hrs
Logical basis of computer structures; machine representation of numbers and characters; instruction codes; arithmetic and logical operations. Techniques of computer programming. Introduction to assembler and procedure-oriented languages (Fortran). Computer applications in science, mathematics, psychology, economics.

CS 006 Introduction to Business Programming 3 hrs
An introduction to computer programming using COBOL. Emphasis on business applications.

CS 107 Computer and Programming Systems 3 hrs
Introduction to assembly systems; monitor and executive systems; structure of languages; compilers and programming techniques. CS 005

CS 109 Computer-Extended Instruction in Mathematics 3 hrs
Intuitive and problem-oriented approach to the calculus using the computer as a tool; computer solutions of linear systems of equations; introduction to linear programming and Boolean algebra; elementary probability. For secondary teachers of mathematics. CS 005

CS 110 Systems Analysis 3 hrs
The designing of computerized systems based on investigations of business systems. Includes documentation and the assisting with implementation.
CS 005, 107, or the equivalent

CS 121 Statistical Methods for Research 3 hrs
Computer analysis of research data; interpretation of statistical tests used frequently in research. Ps 001 and/or CS 005

CS 127 Numerical Analysis I 3 hrs
Solution of equations; polynomial approximations; interpolation, quadrature; error analysis; initial value problems for ordinary differential equation, matrix inversion and matrix eigenvalues. Mt 132

CS 128 Numerical Analysis II 3 hrs
Extension of matrix methods; iterative methods; boundary value problems; finite difference methods; partial differential equations of second order. Mt 132

CS 131 Programming Languages I 3 hrs
Assembler language programming; subroutine linkage; address modification; use of base registers and index registers; byte and bit manipulation; macro definition. CS 107

CS 132 Programming Languages II 3 hrs
Programming in problem-oriented languages, such as, COBOL, PL/1, APL; applications in these languages. CS 107

CS 136 Information Science 3 hrs
The representation, analysis, and retrieval of information. Files processing, data reduction, including data arising from physical scientific experiments, survey data, document preparation, communication systems.
CS 005

CS 141 Programming for the Arts and Humanities 3 hrs
Application of computer techniques for research in the arts and humanities; use of list processing languages and string-oriented languages such as SNOBOL; searching sorting, and pattern recognition techniques.
Junior standing

CS 155 Theory of Automata 3 hrs
Definition of finite automata; description of finite automata; potentially infinite machines such as Turing machines, non-deterministic machines, probabilistic machines; theory of computability.
CS 005, Mt 144

CS 156 Systems Simulations 3 hrs
Computer simulation utilizing logical, mathematical, and Monte Carlo modeling to represent systems; description of the status of systems by the use of sets of entities and the modification of this status by events. Use of special simulation languages.
CS 005, Mt 001

CS 158 Logic and Algorithms 3 hrs
Introduction to Boolean algebra; propositional calculus; quantification theory; algorithms and computing machines; Markov algorithms; Turing machines. CS 005

CS 159 System Programming 3 hrs
The designing of software systems including the use and implementation of assemblers macros, loaders, compilers and operating systems. Advanced approach.

CS 160 Computer Seminar
Current and advanced topics in computer sciences. Five CS courses, Consent

CS 190-191 Professional Development 9-12 hrs
A program which allows a student to spend one semester or one semester and one summer in business, industry, or at a research center as a computer scientist trainee.
Acceptance into program

CS 199 Independent Study
Consent of instructor

HOW TO RUN A COMPUTER CENTER WITH AMATEURS

Robert M. McCroskey
Whitworth College
Spokane, Washington

Introduction

Whitworth College, a liberal arts college of 1300 students, acquired its first on-site computer facility just about a year ago. Prior to that time, we utilized various outside computers for both educational and administrative computing. A course in FORTRAN programming was first introduced in 1968, with student programs being transported across town to Gonzaga University for processing on an IBM 1620. Administrative data processing, consisting of a student accounts receivable system, a primitive registration system and an equally primitive payroll system, all card oriented, were then handled on college unit record equipment.

Three years ago, we converted the administrative systems to computer operation, although still card oriented, and shipped data decks to Washington State University by bus for monthly processing.

During the academic year 1971-1972, Whitworth became a participant in the Pacific Northwest Cooperative Computing Center Project headed by WSU. We acquired an IBM 1050 terminal for remote job entry and processing of student jobs, most of which were programmed in FORTRAN. Four faculty members attended summer workshops in FORTRAN programming and introduced a number of canned programs to students in their various disciplines.

In spring of 1972, our college administration committed itself to the purchase on an on-site computer facility. The main criteria, other than cost, used for our selection of a particular hardware system were:

1. Enable us to program both educational and administrative applications in high level language suitable to relatively non-expert programmers.

2. Provide for the support of up to 16 terminals in a time-sharing environment.

Our selection of the PDP-11 followed with delivery and testing last July. Our first fully computerized system, the student accounts receivable system, went on line last August.

Computer Hardware

The present hardware configuration of our computer system includes the following:

1. PDP-11/20 processor with 28K words (56K bytes) of core memory. No floating-point hardware, but extended arithmetic element for fixed point operations on order.

2. 256K word fixed head disk drive for system software and storage.

3. Three 1.2 million word cartridge disk drives for mass storage.

4. Two dual DEC-tape drives for small file and program storage.

5. 200 card/min Card Reader.

6. 132 column Line Printer.

7. Two VT05 CRT visual display terminals used mostly for CAI.

8. Seven Model 33 Teletypes for general use.

9. Port for remote job entry from off-campus low speed terminal.

Computer Software

Supplied with our PDP-11 were two fundamental operating systems:

1. RSTS-11 time-sharing system for support of up to 16 low speed terminals plus other peripherals utilizing the BASIC programming language.

2. DOS disk operating system, a single user system utilizing assembler and FORTRAN compiler.

The BASIC language used as the time-sharing language is so versatile in its handling of character as well as numeric data and especially in its handling of file structures that we operate under RSTS-11 all the time for simultaneous processing of both administrative and educational applications. Various system programs, all programmed in BASIC, are supplied for system management and computer usage accounting. The use of the high level language for all applications is a boon to the relative amateurs running the computer center. My operations supervisor never saw the BASIC language until the arrival of our computer.

Computer Center Organization

The line organization of the college with respect to computing is as follows:

1. The Vice President/Dean has basic authority over the computing function. In the year we have had the machine, he has hardly set foot in the computer center.

2. The Administrative Assistant to the Dean acts as laison between the top administration and the center.

3. The Computer Services Committee determines overall policy for the computer operation. The committee includes the Administrative Assistant as unofficial chairman, coordinator of computer services (myself), supervisor of operations, the chairman of the modern languages department who was most instrumental in obtaining the computer for Whitworth, and the head librarian in whose building the computer center is located. The particular make-up of the committee was determined by the college president.

4. The Coordinator of Computer Services (myself) determines practical policy for the computing center, acts as system analyst, supervises administrative programming and functions as head programmer all on 1/3 to 1/2 of his official work load. He teaches the rest of the time.

5. The Supervisor of Operations supervises data collection and correction, supervises keypunching and machine operation, and acts as head keypunch operator and head machine operator.

6. There is one fulltime keypunch operator/clerk and a student clerk 10 hours per week during the academic year.

7. Student programmers help with all administrative programming.

Computer Applications at Whitworth

Some of the various applications implemented on our computer during the past year are the following:

1. Use of the computer for general problem solving in various disciplines available to all students who learn some BASIC programming.

2. Computer Assisted Instruction (CAI) implemented by several instructors and described more completely in another section of this paper.

3. Student accounts receivable and general students records systems implemented during the first year of operation. Other administrative systems such as college development gift records, alumni records, admission records, etc. are being programmed this summer.

4. Elementary computer fundamentals introduced through computer science courses.

Computer Science Instruction

At Whitworth we are not trying to compete with the university computer science programs. Rather, we are providing basic instruction in computer fundamentals and programming to enable the student to use the computer for problem solving in his chosen discipline or move easily into a computer science program upon graduation or transfer to a university. At present we are offering the following courses:

1. Introduction to Computing (BASIC language)

2. Computer Processing in the Sciences

3. Computing in the Humanities

4. Introduction to computer Organization and Data Structures

Exposure for a selected few students to advanced topics in computing is handled through independent study. It is our hope that the computer will be more and more utilized by many disciplines and that computer instruction will not be left to the so-called computer specialists.

Computer Based Instruction at Whitworth

While the typical approach to computer assisted instruction (CAI) is to develop software capability to approximate as nearly as possible the intelligence and flexibility of a living instructor, the philosophy of the Whitworth CAI system has been much less ambitious, less costly and better able to meet specific needs. The critical portion of our procedure has been for the area specialist (the instructor) to determine which aspects of his instruction are already mechanical or routine in current classroom practice. The logical tactic, therefore, is to make that portion of the instructor's effort more widely accessible to his own students and interested outsiders.

We set as our first priority the development of an author language whereby area specialists themselves might implement original curricular material without they themselves being programmers. That package, developed by Dr. Ron Turner, known as WHITCATS ("Whitworth Computer Assisted Tutorial System") is currently being used in the areas of library orientation and research techniques, English literature, church history and philosophy, general study and test-taking skills and foreign languages.

In the first-year Spanish course, the CAI material constituted the entire written exercise material. Written evaluations by participating students were highly favorable. In addition to the coursewriter language, specific programs, all written in BASIC, are used constantly by foreign language students for drill and tutorial work utilizing the special graphics capabilities of the CRT terminals.

Although we plan to implement a record-keeping facility for true instructional and test-generation capabilities, we intend always to keep the lesson programming as simple as possible and within the grasp of the area specialist.

Our next significant software development will be a graphics package to offer a variety of text formats and illustrative capabilities to many areas of the college curriculum.

Administrative Applications

The BASIC language has been utilized to implement several administrative file structures on 2.4 million character disk cartridges. Virtual core storage is used to define data in lists and tables on the disk. Thus, a prototype file structure will consist of three basic files (1) a KEY file which contains all ID numbers of persons active in the file, (2) a DATA file which contains an alphameric record for each person in the file, and (3) a LINK file which contains linkage between KEY and DATA files for each person in the file.

Descriptions of files are handled easily in a program by means of OPEN and DIMension statements in the following manner:

```
10 OPEN "FILE.KEY" AS FILE 1
20 OPEN "FILE.LNK" AS FILE 2
30 OPEN "FILE.DAT" AS FILE 3
40 DIM #1, K(3000)
50 DIM #2, L%(3000)
60 DIM #3, A$(3000)=256
```

File "FILE.KEY" contains a list of no more than 3000 floating-point values (ID numbers). File "FILE.LNK" contains a list of no more than 3000 integer values (links). File "FILE.DAT" contains a list of no more than 3000 character strings (data records) each of which is exactly 256 characters long. To access a particular data record requires the position of the ID number in the list of "FILE.KEY" and the link in the same position of list of "FILE.LNK". The appropriate data record in the list of "FILE.DAT" is assigned to a string variable. Example: B$=A$(L%(I%)) where I% gives the position in the key file. Many programs, each under 8K words in length, are used to create, delete, maintain, and report from file records, all programmed in the high level BASIC language. Included is a sample program which prints out a summary listing from a student records file structure:

```
1000 REM ** INACTIVE FILE LISTING PROGRAM
1002 PRINT "INACTIVE FILE LISTING PROGRAM"
1004 INPUT "WHAT DISK ARE YOU USING" ; N$
1010 OPEN N$+'STUDDM.KEY' AS FILE 7
1020 OPEN N$+'STUDDM.LNK' AS FILE 8
1030 OPEN N$+'STUDDM.DAT' AS FILE 9
1040 DIM #7, K(20000)
1050 DIM #8, L%(20000), P%(10)
1060 DIM #9, A$(20000)=64
1070 P1%=P%(1%)-1%
1080 OPEN "LP:" AS FILE 6
1100 K%=80% : F$='#.## ##.## ##.## ###.##'
1110 GOSUB 2010
1120 FOR I%=1% to P1%
1130 B$=A$(L%(I%))
1150 C$=LEFT(B$,5%)+' '+MID(B$,6%,25%)+' '
1160 C$=C$+MID(B$,31%,3%)+' '+MID(B$,34%,2%)+' '
1170 C$=C$+MID(B$,36%,1%)+' '+MID(B$,37%,1%)+' '
1180 B1=CVT$%(MID(B$,42%,2%)) : B1=B1/100
1190 B2=CVT$%(MID(B$,44%,2%)) : B2=B2/100
1200 B3=CVT$%(MID(B$,46%,2%)) : B3=B3/100
1210 B4=CVT$%(MID(B$,48%,2%)) : B4=B4/100
1220 PRINT #6, C$;
1230 PRINT #6, USING F$, B1, B2, B3, B4
1240 GOSUB 2010
1250 NEXT I%
1260 GOTO 3000
2000 REM ** PAGE HEADING ROUTINE
2010 K%=K%+1%
2020 IF K%<60% THEN RETURN
2030 PRINT #6, CHR$(12%);DATE$(0%);SPACE$(10);
2040 PRINT #6, 'STUDENT PROFILE INACTIVE FILE LISTING';
2050 P%=P%+1%
2055 PRINT #6, SPACE$(10);
2060 PRINT #6, USING 'PAGE ###', P%: PRINT #6
2062 PRINT #6, SPACE$(44%); 'GPA  ATT  PASS  GP':PRINT#6
2065 K%=4%
2070 RETURN
3000 REM ** EOJ
3010 CKISE 6,7,8,9
3020 END
```

Problems in Computer Center Operation

Naturally, we have encountered our share of problems in the operation of our computer center. Some of these are:

1. A lack of understanding and even an unwillingness to understand overall personnel and programming needs of the center.

2. Inaccuracy of source data supplied by other departments for administrative applications and even an unwillingness on the part of some to be responsible for the accuracy of data.

3. Lack of sufficient personnel to handle the increasing operation and programming load of the center. Coordinator becomes supervisor, supervisor becomes operator, etc.

4. Difficulty in meeting deadlines imposed by others desiring service from the center. Expectation on the part of some for "instant" service.

5. Difficulty in getting other departments to analyze their operations and communicate their needs effectively.

6. Lack of support for the concept of a computer on campus by those, faculty and student, who consider it a threat or a menace.

NETWORKS IN WASHINGTON STATE

William E. Walden
Washington State University
Pullman, Washington

The title of this section of the program is entitled Where Are We Now. I noticed that in earlier presentations there is apparently considerable acceptance of the idea that the small college can use networks to obtain its computer services. Perhaps I can say then that is where we are now. I recall that in 1967 I, like others, was concerned that students in small public and private colleges in Washington did not have access to the kinds of computer services available to university students. I visited several small colleges in the State of Washington and suggested the possiblity that they could obtain services via remote terminal from Washington State University. The response, with the exception of two institutions, was not at all enthusiastic, and in fact quite negative in most cases. Now, today, there are twenty-three institutions acquiring services remotely from Washington State University. Today I am going to present my view of networking in Washington as of now.

During the past two years I have been involved in two projects that are having an influence on networking in the State of Washington. One is a rather extensive network study sponsored by Higher Education, and the other is an existing network with central computing facilities at Washington State University. This report briefly describes these efforts and discusses various resulting conclusions.

In May 1972, Presidents from the four state colleges and the two universities, together with the Community College Board, authorized the joint expenditure of funds to conduct a network study to determine the feasibility of an educational network in the State of Washington. As a part of the study, three knowledgeable consultants visited the State of Washington and helped relate our situation to other states and networks. Staff, hired for the study, visited fourteen networks throughout the United States. Through reports and/or discussions with individuals, many other networks were studied by staff members. Thus, in one way or another, 29 existing or proposed networks were examined. Those examined included statewide educational systems such as the Georgia University System Network, included consortiums such as the Associated Colleges of Central Kansas, included single universities with branch campuses such as Pennsylvania State University, included universities that have simply extended their services to other institutions such as the University of Iowa, included the universities that have combined to form non-profit corporations such as the Triangle Universities Computing Center, included profit making corporations totally owned by universities such as the CAI Corporation, and included existing and proposed national networks such as ARPA and the National Education Computing Service. The staff also tried to develop a complete bibliography on networks and attempted to review and summarize every publication in the bibliography. These reviews do not appear in the Network Study Report but were available to all individuals involved in the study.

The committee responsible for the study made the following recommendations, precisely quoted:

1. We suggest that the Council of Presidents examine major questions of educational philosophy brought about by plans to share educational facilities (agreements on priority, contribution, organization, control, budgeting).

2. The Council of Presidents should continue the existing Network Committee to review progress toward resource sharing. This committee should make recommendations for further implementation at an appropriate time. This committee should prepare forecasts of requirements and necessary resources to fund meeting those requirements on shared facilities.

3. Whenever computing requirements exist beyond the campus capability, the Community Colleges should utilize, via terminals, a University, State or Community College computing facility for academic and administrative computing.

4. Every state college and university should be provided with supplemental funding to install, and use, a Card Reader/Printer Terminal or equivalent capability which can be used to access available resources.

5. Each School District and High School should be encouraged to use the terminal facilities at the institutions of Higher Education.

As a result of this study I have reached some conclusions which I want to present here. I emphasize the fact that these are my conclusions and not necessarily the conclusions of the entire group that worked on the Network Study Report, although this group agreed on many of the statements which do appear here. Some of the conclusions may appear to be so obvious that they do not need repeating. However, I have concluded that many unusual statements about networks have been made in the past, so perhaps some of the conclusions are not so obvious. The conclusions are:

1. Administrative and academic computing services can, and are being provided via remote terminals from distant central facilities.

2. Greatest network success occurs when a well-managed, existing center is expanded.

3. The most substantial savings are in the avoidance of duplication of programs and systems.

4. Communications problems can be satisfactorily solved for networks.

5. A large, central computer provides a variety of services not possible on smaller computers.

6. There are several successful networks in the United States, and several unsuccessful ones.

7. Technical problems are usually due to poor planning and can be solved.

8. Most failures of networks are related to poor planning and improper management rather than organization or location.

9. There must be cooperation either through incentives or strong central control in order to have a functioning network.

Another project which is influencing discussions of networking in the State of Washington is a network with central facilities at the Washington State University Computing Center. This network received supplemental funding from a National Science Foundation grant in 1971. The purpose of the grant was to provide faculty training in the use of computers but in so doing some funds were provided to each user institution for equipment and computer usage. There were 11 institutions tied into the Washington State University facilities as a result of this initial grant. However, the network has now grown to 23 institutions of various kinds. The network is primarily serving Washington institutions although there are three out-of-state institutions participating.

Distances of user institutions from Washington State University vary considerably. The minimum distance is 2 miles for Pullman High School and the maximum distance is 280 miles for Evergreen State College. The number of schools associated with certain distances is:

less than 20 miles	2
50 miles	3
100 miles	7
200 miles	18
300 miles	23

Types of services provided in the network are:

1. Interactive languages including CPS PL/1, CPS BASIC, Computer Assisted Instruction, and an Abstract Retrieval System.

2. High-speed remote job entry with availability to all batch resources.

3. Low-speed remote job entry with availability to all batch resources.

In the network, terminals being used for low-speed remote job entry are the IBM 1050, IBM 2741 compatible terminals, and Model 33 teletype compatible terminals or displays. Terminals in the network which are being used for high-speed remote job entry are IBM 2780 compatible terminals together with the following computers: IBM 1130, IBM System 3, UNIVAC 9200, Systems Engineering Laboratory 810-B, Raytheon 704.

The network centered at Washington State University has turned out to be an experimental laboratory for the State of Washington. This is by virtue of the fact that many different kinds of institutions are being served, many different kinds of services are being offered, and the services are being utilized for various types of processing. Private education, public education, and government agencies are represented among the user institutions in the network. High Schools, two-year colleges, four-year colleges, and extension centers are included in these user institutions. Some institutions do all of their academic processing via remote terminal. One institution does all of its administrative processing remotely. Most of the user institutions are using the network for some of their academic processing.

So far the workload from institutions other than Washington State University can be characterized as many jobs with low CPU requirements; that is, student type jobs rather than research. This is reflected in the fact that the average cost of usage by institutions external to WSU is $8,000 per month whereas the WSU usage was $87,500 per month. Yet the average number of jobs for external institutions was 5,000 per month while the average number of Washington State University jobs was 21,000 per month.

Now my conclusions based on the network with central facilities at Washington State University are given:

1. A campus can do all of its academic and administrative computing via remote terminals to a central facility.

2. A small college should use an educational facility rather than a commercial or state agency facility.

3. There has been a considerable interest in compilers, programs, and systems which cannot be run on a small computer.

4. Training of faculty has been most successful on those campuses which have at least one person with computing responsibility.

5. Technical problems can be, and have been solved by Washington State University staff.

6. The central facility must have staff dedicated to assisting other campuses in solving technical problems.

7. The network makes transfer of students more easily accomplished.

8. The network can be set up so that the central campus is not a favored customer.

9. Such a network greatly increases faculty interchange and sharing between campuses.

In summary, it would appear that the acquisition of computing services via remote terminal is a feasible alternative for colleges as of now. The college which is considering such an approach should thoroughly investigate the central facility from which services would be obtained. It should be well managed, have a long history of operation, experience with terminal service, and preferably be an education based center. Also, staff should be available to assist remote institutions with problems related to use of the central facilities. If this alternative is selected, then the college should have one person responsible for use of the terminal, just as they would have a person responsible for a stand-alone computing facility.

USER SERVICES - THE STATE OF THE ART AND ITS IMPLICATIONS FOR SMALL COLLEGE COMPUTING CENTERS

Susan Kolasa
Stanford University
Stanford, California

In April of this year it was my privilege to chair a national conference on User Services in University Computing Centers sponsored by SIGUCC.

The steering committee planned technical sessions which we hoped spanned the activities and interests of those involved in this specialty. Included were two sessions designed to appeal to small college representatives -- "Special Problems of Small Colleges" and "User Services in the Network Environment".

This conference was the first such undertaking and we were not sure of its reception. For planning purposes I estimated 50 attendees. 200 people registered from all parts of the country including Puerto Rico, and several from Canada. Attendance at the closing session was as great as at the first. There were even "birds of a feather" ad hoc get togethers scheduled by attendees in the evening.

Such interest and concern with the user interface has not always been the case. The computing center of the mid-60's when I entered User Services, though already a service facility, had a vastly different level of concern for its users.

The user located the resource person, usually a systems programmer at his (the programmer's) convenience. This was normally between 10 p.m. and 2 a.m. in the machine area where the programmer, his peers, friends and occasionally supervisor, were occupied with hands on machine time, studying listings or conversing in jargon which might or might not have been business related. Central to the scene were one or more machines blinking lights, clattering messages, spewing paper. It or they appeared in command. The programmer was usually tired, harassed, unkempt and needing "just one more run".

The user who could get attention did so by waiting until it was convenient for the programmer to break off his other activity. He kept attention by having an interesting problem -- from a system, not the application point of view. If he had found a genuine bug or was trying some interesting technique, or was noisy enough about seeing he got help with his problem, he would be assisted. He might get instant satisfaction or a firm promise, rarely noted in a log, to fix the error. Otherwise he was shortly aware that his problem was too trivial to bother the programmer who would return to his interrupted task, sometimes after remarking audibly upon the level of the user's competence.

But faculty have a way of communicating effectively, and soon there would be a new service announced -- User Consultants. I can still see an employment notice. "Must know FORTRAN, 4 consecutive hours minimum a week, $1.50 an hour." What the ad didn't say was that those hours would be spent in an underventilated classroom refitted with keypunches, interpreters and sorters, tables for user set-ups and a place to queue up for the consultant. It also failed to mention that the consultant would be asked questions on plotting, assembler, JCL, utility programs as well as FORTRAN. What the users didn't realize, at least initially, was that the consultants, all busy undergraduates, did their best for the shift and then disappeared from the center. They were loosely coordinated by a supervisor who could never find a time when all could attend a meeting. Thus notice of changes in the system or policy rarely reached them ahead of the users, and the quality of their advice suffered accordingly. Since turn-around time was slow, a problem answered by the consultant would not be tested until he was off duty. His replacement frequently had conflicting answers to give and the user, as they used to say in victorian novels, "retired in confusion".

We have come a long way in these few intervening years as witness the User Services Conference. The most dramatic demonstration was a slide presentation by Dr. Ronald Rutledge, Director of Carnegie-Mellon University's center. He showed the group pictures of a bright, carpeted user area which is connected by closed circuit TV and 2-way audio communication to the machine room 24 hours a day. A user could hold a listing up to the camera and be advised of his error by the system personnel. He sometimes received help before he knew he needed it. The monitors over the user operated card readers would detect an improper approach to using the device. Out of the ceiling would come a voice, "Please turn the cards over before inserting them". Ron said this could be quite a shaking experience the first time.

We heard presentations that indicated the state of the art in face to face consulting. Whether by graduate student consultants, e.g. University of Michigan, or full time employees with a range of degrees through Ph.D. across a spectrum of user specialties, e.g. Stanford, the consultants are finely screened for competence before hiring, are trained further on the job and are taught to say, "I don't know, but I'll find out", when the situation requires it.

User education is no longer a grapevine or inner circle exchange of information, but a major part of user services' function. As Bruce Lemm, Manager of User Services at Stanford and chairman of this panel pointed out, education is offered in several ways. Normally one thinks of this in terms of course offerings. Many centers do provide short (2-12) session courses in the languages and systems available at or through their installation. Other forms of education include seminars for special interest groups, e.g. social scientists. These serve two worthwhile ends. First, they relate information which is specific to the needs of a discrete group; second, they can be an excellent marketing tool. This is equally as valuable to a department wishing to demonstrate its need for a private terminal or special software package as to the center interested in increasing usage.

Documentation, the topic of another session, also fills an educational need. We found that good documentation in the form of updated user manuals, newsletters, vendor and installation prepared tutorial and reference materials is generally available. Increasingly some or all installation documentation is being kept on line in user accessible files. Further documents such as newsletters are frequently exchanged among installations so good ideas are picked up and implemented across the country.

Other functions discussed included: input/output methods, user accounting, and personnel selection. A growing number of installations feature user operated card readers and printers to minimize turn-around and staff handling.

User accounting is generally quite sophisticated and detailed, telling the user his costs per job and in some institutions summarized for him daily. Management reports on unit costs are also generated for the installation directorate so that accurate internal accounting can result in accurate cost recovery procedures.

Again, generally the center sees the user as a paying customer whose dollars are allocated by sources external to the center itself. Gone are those harrowing days of a center user services manager or director faced with allocating resources between two researchers or instructors each with the same deadline and a valid utilization. Now users may buy priorities in addition to other services.

Personnel selection and training was a workshop session chaired by Dr. Ted Willoughby of Penn State University. It was designed to get us all thinking about how to properly evaluate performance and initial selection criteria. A three step procedure was suggested. First, try to describe some specific behaviors which argue for effective, ineffective, or moderate performance of the employee. Second, cluster analysis behavior into meaningful clusters. Third, assign scores to behaviors.

As I mentioned at the outset, two sessions, networks and special problems of small colleges were particularly directed to the small to mid-sized installations.

Dr. Fred Weingarten reminded us that small colleges are largely undergraduate, usually privately run, have no more than a few thousand students, and are teaching rather than research oriented.

This and the network session provided a picture of the types of computing alternative approaches utilized a) stand-alone centers, b) consortia, c) terminal or couriered service from a larger central site.

In each case, implicitely if not explicitely, the bulk of the user interface is provided by on site staff. Last year's conference in Atlanta reported that a small installation may operate on a budget of $20,000 to $25,000, and that a medium sized budget tops out at $500,000 for <u>all</u> center expenses. The large installations on the other hand, may spend up to $200,000 or more on user services alone.

In the smallest installation the entire center staff may be one 1/2 time faculty member. The large installation may have 10 or more full time professionals in user services.

One can, of course, argue the distinctions. A large unit may serve 5,000 or more local users and a network of remote users from other educational institutions. There is usually a heavy research population requiring a wide range of sophisticated service and at the same time a high utilization by undergraduates locally and remotely for WATFIV, etc., as opposed to the small college needs previously described.

However, on balance, economies of scale clearly favor the large centers. Their research users are frequently very knowledgeable or have programmers on their own staff. Similarly many class users are from engineering, math or computer science -- computer oriented disciplines. Lastly, there are staff members available to specialize in skills to aid less typical applications, e.g. social sciences, humanities.

In contrast the 1/2 time faculty member must be all talents to all users. He must be consulting, education, specific application documentor, hand holder and salesman in addition to the other functions of director, emergency maintenance, operations and accounting.

The director or coordinator can react in one of three ways to this assignment. He can let the users fend for themselves as he attends to other priorities, he can try to do everything his various "hats" entail alone, or he can be inventive, accomplishing the tasks without ruining his own health.

The most successful people, of course, are those in this last category. The tips they passed on at our conference and through my observation of their activity are straightforward and effective.

1. Utilize your vendors. They can put you and your faculty in touch with faculty at other installations doing similar projects thus creating user self reliance. They can give seminars and courses for you. This is true of hardware and software suppliers of an inhouse installation and of the source of remote computing service. The point to bear in mind is that your satisfaciton with them is a vital ingredient of their business success.

2. Involve people in the user community in helping provide service to others. A very clever idea -- which actually came from UCLA which is not a small installation, is to encourage a student computer club. Provide its officers with N hours of computer time in exchange for consulting or teaching programming courses. Insure your satisfaction by requiring a stated level of performance for the computing time, and place the burden of quality control on the officers.

3. Encourage other faculty to serve as the focal point in their departments for discipline oriented computer services.

4. Keep in touch with peers through newsletter exchange and meetings such as this one to learn new ideas and techniques.

5. Keep your sense of humor.

In summary, User Services has developed over a short span of time into a highly professional computing specialty. It has become a necessary ingredient of every educational service facility's staff regardless of size of the facility or source of its computing power. Whether performed by a discrete staff or as part of one person's total responsibilities, it is a vital link between the center and its users.

SECTION II

WHERE ARE OUR STUDENTS GOING?

WHERE ARE OUR STUDENTS GOING?

Joyce Currie Little
Community College of Baltimore
Baltimore, Maryland

Introduction

In any discussion concerning an educational endeavor, it is customary to define the objectives of that endeavor. After acknowledging whether or not the objectives have been fully met, it is hoped that we are able to find out why they have not been met, and determine what we can do to improve the situation. From the conglomeration of programs offered currently by two-year colleges ranging from key-punch training, operator training, data processing -- both business and scientific options, and computer science, I'd like to center my remarks about only those students who enter the data processing curriculum. The objective of this program is to train applications programmers, in either a business or scientific option, but with the overwhelming majority in the business option, planning to become commercial applications programmers.

In one fifteen minute talk, I do not intend to attempt to cover any topic other than that -- for more information on what we teach, who we get as students, where we get teachers, what equipment we have, what our curriculum is like, what problem sets we've developed, what textbooks we use, what service courses we have, what articulation problems we've had with other colleges -- please feel free to contact me later. Or perhaps you'd like to join the two-year college people presently joining ACM's Special Interest Group in Computer Science Education, or the Association for Educational Data Systems (AEDS). Neither will I attempt to discuss where students in our service courses are going, even though there have been many instances of these graduates entering the computer professions.

Where Do Our Students Go?

Believe it or not, some of the graduates actually begin as programmers, even though most must undergo a training course, whether they need it or not. Others, for a variety of reasons, fail to get programmer jobs, but find entry-level work as computer operators. Many drop-outs also find good jobs -- some as operators, some even as programmer-trainees.

Some graduates go into non-computer types of work, such as real estate, construction, insurance, or accounting. Almost 40% of our graduates go on to take more courses, often toward a four-year degree. Most, however, go on in evening school rather than day school, while working full-time in data processing. Most of those who go on to school major in Data Management, Business Administration, Business Data Processing, or Information Systems Management. None have yet gone into the 4-year Computer Science program; although a few have had sufficient background for admittance, there has not been a program of that type near us.

Some female graduates get married, have children, and do not work in the computer field at all; a few others in that situation work part-time as programmer-analysts. Very few graduates leave the urban/suburban area, other than those several who choose to work for the U.S. Air Force, in data processing. One student has gone to Israel to look for work; one other took a job in Washington, D.C.; another one took a job in New Jersey.

Many students taking jobs as operators are with banks or small companies, but most of these are with U.S. government service, with the Social Security Administration (SSA). Arrangements were made by our college for summer work for students between their freshman/sophomore years; in later years, SSA extended this policy to include community colleges in Baltimore County as well. Many students return to these jobs after graduation, even though they could in many cases get into programming sooner elsewhere.

Graduates in the scientific option, or those with more mathematics or electronics electives, go into data analysis of a statistical nature, in research centers at universities or medical schools. Others go into process control work with engineering companies. A small number of graduates have become assistant instructors, working as laboratory assistants in community colleges.

What Types Of Companies Hire Them As Programmers?

Graduates with the Associate in Arts degree in Business Data Processing are hired as programmers by a variety of companies. More than half the graduates who start as programmers are in banks, savings and loan companies, financial or insurance companies, service bureaus, and data processing departments for city and state agencies. Many others go into computer center work in educational institutions or hospitals. A few go to one of the several large nationwide network facilities; a small number -- the more mature and more versatile -- go into programmer/analyst positions with small local companies.

What Is The Career Path Of The Graduates?

Most of those taking jobs as operators progress in one or two years to the programming staff; operators with the federal government, however, take as many as 3 or 4 years, sometimes longer, to progress that far. Those who started as programmer trainees become programmers rapidly -- usually in less than 3 months. Those in large companies tend to specialize -- one student is doing in-house training, one is doing software development, one is in systems analysis for new on-line applications. Those in small companies tend to be doing everything, and can progress in 3 to 5 years to managerships. Several who started as programmer in educational institutions are now performing special computer services for top executives or administrators. Others have developed interest and expertise in systems programming, doing development of job control accounting systems, data base development, and operating systems design.

The fairly rapid advancement of these graduates opens the door to another graduate to replace him. Therefore, jobs are not only available in those areas of new applications, in those companies just beginning to convert to computer usage, in companies expanding their computer applications but from the turnover created by the movement up the career path ladder.

What Prevents Students From Getting Better Job Opportunities?

1. The business community is not even yet fully aware of the value of the data processing graduate. After an initial good experience with a student, a company will often call to request applicants for their next job opening. Most companies, however, still require a training program of all entering programmers -- not just training to acquaint the student with the methods and procedures in use there that may be unique to that compnay, but a full-fledged in-house training program from scratch, starting with what the holes mean in the cards! Graduates would, of course, prefer to take their expertise to a compnay ready to recognize his background and build upon it, rather than to repeat part of this training.

2. Some companies hope to maximize their chance of getting a "good" product by hiring a four year college graduate, no matter what their major, and training him themselves, rather than take a chance on hiring a two year college graduate.

3. Many companies require experience, no matter what kind. Programming graduates who have bagged groceries over their two year college career have an advantage over those who have never been employed.

4. The custom of civil service hiring for government employment encourages promotion from within; graduates of 2 year colleges in our area must start as peripheral equipment operators, progress to console operators, then apply for programmer training class, hoping to be selected; they then have little difficulty excelling over other trainees who have often had no former training in programming languages. In spite of this difficulty, students go eagerly into civil service work -- not only is the pay for these entry level jobs good, but the work is easy, the environment is pleasant, the job is relatively secure, and there is that opportunity, albeit time-consuming, for advancement.

5. Graduates often lack other attributes felt necessary for the job; for example: many score low on vocabulary exams; many lack the maturity necessary for the job; many make a poor impression during an interview. Others have little regard for the demands of the business world toward mode of dress or appearance. Students have been known to refuse a job rather than abandon their poncho and beard!

6. Personnel men for some companies have notoriously little expertise in evaluating the qualifications of applicants in the computer field. They have been known to allow their limited knowledge -- often evident in their use of certain trite phrases -- to sway them against a community college applicant. An example is: Although you had COBOL on a large 3rd generation UNIVAC system, we are looking for someone who has used COBOL on a system exactly like the one we have. Another: Your transcript says you had work in punch card systems; we use magnetic tape systems. And yet another: Your hands-on experience was a 2nd generation computer; we want someone who has had hands-on work with a 3rd generation computer. Unfortunately, before an interview can be set up with those persons in data processing departments who know better, the student must somehow get through the personnel officer.

How Can We Improve Quality Of, And Placement For, Our Students?

1. Acquaint community businesses and institutions with curriculum, student populations, equipment, etc.

2. Choose advisory board members from the variety of businesses you hope students will serve.

3. Encourage improvement of curriculum and faculty to keep applications and techniques taught up to date.

4. Work toward better articulation with the four year colleges to facilitate easy transfer of courses.

5. Encourage some selection of the students applying to enter your program by attempting to acquaint high school counselors with the needed skills, traits, personal characteristics desirable in the field, and with the demand for workers in the field at the time.

6. Encourage the development of certification examinations for the different job levels; promote the Registered Business Programmer Examination (RBP) for your business applications programmers.

7. Arrange work/study programs, with proper supervision, to put students into a company for work experience on the job, either for credit, or for pay.

8. Offer free computer time to companies without computing equipment, to allow them to investigate computer usage with the help of a student; have them pay the student on a temporary, part-time basis for doing one project. (Often the student will get hired and the company will get equipment.)

9. Give students a chance, outside of class, to learn more about what is expected on the job, by visiting alumni now employed, visiting companies to talk to programming staff, visiting computer exhibit areas to talk with representatives.

10. Offer students the chance to learn the attitude of loyalty and service to their employer by sessions simulating problems that may arise; offer them the chance to practice for the interview by having interviews on campus, or with each other; encourage them to show responsiblity and maturity in part-time work and in class activities. (Some of these can be done by means of a Computer Club.)

How Can We Find Out More About Our Graduates?

Many colleges have no idea what happens to their graduates; other ask, via surveys, only a few vital questions, such as those concerning salary. Others have alumni groups, with a newsletter offering news of alumni from all the disciplines. It would be more helpful to have special alumni meetings, specifically for this field, in order to enable graduates to return for special programs, for professional development seminars, for advising current students, for acquainting themselves with other graduates, and most importantly, to provide feedback information on the effectiveness of the curriculum. The extra time it takes to do this reaps untold rewards in maintaining a viable, workable, flexible curriculum from year to year.

COMPUTER SCIENCE PROGRAMS IN CALIFORNIA

Curtis P. Gerald, Professor of Computer Science
California Polytechnic State University
San Luis Obispo, California

Kenneth M. Tom, Director of Computer Center
California State University
Long Beach, California

Computer science education comes in many shapes and sizes, and current debate (the Kendal-Wegner controversy and more recent upwellings[1,2]) highlights the diversity of points of view. However, the identity crisis that has beset our discipline from its beginning is not unusual in a new subject matter area; the present arguments over just what we should mean when we say "computer science" can only help to clarify issues that are important to higher education and vital to a society that is more and more dependent on computers.

In part, the many applications of the computer itself contribute to our confusion. The words, "data processing," are used by some in a generic sense; to others, they connote business-oriented uses only. The scientific community has had so much need of the computer in conducting research that only recently has there been time to develop its applications in the social sciences. The origin of many who call themselves computer scientists in mathematical disciplines has given them a definite bias. At the same time, computer systems have evolved into general purpose machines and specialization is an attribute only of the people-ware, not the hardware nor much of the software.

But applications themselves, though of tremendous importance, are not the central theme. For most of us, the words "computer science" should have meaning apart from computer applications (though not divorced from them). But computer systems have changed so fast that the thing we are concerned with is nebulous and uncertain. The interface between the hardware and software has continually shifted and the areas of interest have widened as our industry has grown and found alternative solutions of greater sophistication. It is hoped that this paper, surveying the spectrum of undergraduate curricula available in California, will help to clarify the situation. This paper does not attempt to describe programs that lead to the Master of Science or Doctor of Philosophy degrees.

Curriculum 68 is usually the starting point for any discussion of computer science programs. Many of the earlier programs were modeled after this pattern, but for at least a substantial number of institutions, this path led to disillusionment. The comments of George Gorsline and Duff Green are typical:[3] "Curriculum 68 came under early and continuing heavy criticism from within and without academia for its almost total neglect of the pragmatics of the job market." They then describe how their program, initially relying on Curriculum 68 for its pattern, was redesigned to recognize and accommodate the needs of industry and government as employers of their graduates. Many of the more successful programs now describe themselves by the ways that they differ from Curriculum 68.

In the State of California, there are more than 80 private institutions of higher education, 19 State University campuses, 10 University of California campuses, and over 90 community colleges.

The various computer science programs available to students in California colleges and universities represent almost all parts of the diverse interpretations of the term "computer science." In the two year community colleges, the typical objective is to prepare EDP applications programmers who enter industry directly, although many of their graduates (with the associate degree) transfer to four-year institutions to complete a more thorough academic preparation. The four-year colleges do not fall into any one design; some are business applications oriented, some emphasize systems programming, and others have strong hardware involvement. Most of the programs in the state universities and some of the private schools anticipate that most of their graduates will enter employment directly after graduation, while the programs in the University of California system concentrate on a good preparation for further graduate study. Because of these differences in objectives, there are significant differences in the number of courses in the areas of computing theory and structure of languages.

There is no established way to classify these different approaches to the teaching of computer science. One way is to distinguish on the basis of hardware versus software emphasis and by the field of application. We might then have:

Computer Engineering -- a program with special emphasis on computer architecture and the implementation of memory, registers, and control functions.

Software Engineering -- a program whose central concern is the development of programs for systems control, resource allocation and scheduling, language translation, utility functions, and user-oriented software.

Scientific Applications -- a curriculum designed to attack problems in the fields of engineering and science, emphasizing numerical analysis, information theory, computational efficiency, and error analysis.

Business Applications -- a set of courses chosen with their relevance to information storage and retrieval, report generation, and management decision-making.

While the above calssification scheme is far from perfect, it shows the diversity of programs in the various institutions of higher education in California because there are curricula representing each orientation. (Some schools have programs extensive enough that the student can choose from several different programs.) Based upon a sampling from the various segments of higher education in California, we show typical curricula of each type. The survey was far from complete. It is contemplated that a more definite study will be made of computer science programs in California through the Interest Group for Teaching Computer Science in the California Educational Computing Consortium.

Computer Engineering

University of California, Berkeley -- Department of Electrical Engineering and Computer Science and Engineering program.

FRESHMAN YEAR

	Fall Units	Winter Units	Spring Units
Math 1A-1B-1C, Calculus, series, and functions....	4	4	4
E1, Computers and Their Applications.....	4	-	-
Physics 4A-4B, Mechanics, Electricity.....	-	3	3
Electives.....	7	8	7
	15	15	15

Recommended Electives: Chemistry 4AB or 1ABC; Engineering Graphics; English or Rhetoric; History, Social or Humanistic Elective Courses.

SOPHOMORE YEAR

Math 51A, Linear Algebra.....	4	-	-
Physics 4C, Waves and Oscillations.....	4	-	-
E41, Programming Languages and Techniques.....	-	4	-
E17, Introduction to Electronics.....	-	-	4
Electives.....	7	11	11
	15	15	15

Recommended Electives: E45; Math 51B, 51C; Biology 1ABC; Physics 4D, 4E; English, History, Economics, Accounting; Social or Humanistic electives.

JUNIOR YEAR

EECS 150, Logic Design and Computer Components...	4	-	-
EECS 152A-B, Computer Systems.....	-	3	3
EECS 151A, Memory and Storage Devices.....	-	4	-
EECS 107, Programming Techniques and Data Structures.....	4	-	-
EECS 118, Fundamentals of Discrete Systems.....	-	3	-
Statistics 134A, Probability.....	-	-	3
Electives.....	7	5	9
	15	15	15

SENIOR YEAR

Electives.....	15	15	15
	15	15	15

Total Units 180
Total Electives 117
Social Science and Humanities 27 (required electives)
Electives are recommended in Computer Components, Computer Systems, Computer Theory, as well as breadth electives

Software Engineering

California Polytechnic State University, San Luis Obispo -- Computer Science and Statistics Department, Computer Science program.

Freshman	Fall	Winter	Spring
Fortran Programming (CSc 101)	1		
Boolean Algebra (CSc218)		3	
Analytic Geometry and Calculus (Math 141, 142, 143)	4	4	4
General Physics (Phys 131, 132)		4	4
Freshman Composition (Engl 104, 105)	3	3	
Freshman Composition (Engl 106) or Technical Writing (Engl 219)			3
Biological Sciences	3		
Health Education (PE 107)			2
Philosophy	3		
Physical Education Activity (PE 141)	1/2	1/2	1/2
Electives	2	2	3
	16 1/2	16 1/2	16 1/2
Sophomore			
Linear Programming (CSc 219)		3	
Computer Principles and Programming (CSc 221)	3		
Digital Computer Symbolic Programming (CSc 222)		3	
Digital Computer Programming (CSc 304)			3
Programming (CSc 310 or 340)			3
Numerical Linear Analysis (CSc 331)	3		
Analytical Geometry and Calculus (Math 241)	4		
Differential Equations (Math 242)		4	
General Physics (Phys 133)	4		
Principle of Economics (Ec 211, 212)		3	3
Literature			3
Basic Accounting (Actg 131, 132)		3	3
Physical Education Activity (PE 241)	1/2	1/2	1/2
Electives	2		1
	16 1/2	16 1/2	16 1/2
Junior			
Advanced Fortran Programming (CSc 301)	2		
Introduction to Numerical Methods (CSc 332)		3	
Numerical Analysis (CSc 333)			3
Data Structures (CSc 345)	3		
Systems Analysis (CSc 350)		3	
Algorithmic Compilers (CSc 351)			3
Statistical Analysis (Stat 321, 322, 323)	3	3	3
Literature or Philosophy	3		
American Government (Pol Sc 201)		3	
General Psychology (psy 202)	3		
Managerial Accounting (Actg 301)			4
Analog Computer Techniques (EL 313)		3	
Principles of Digital Computers (EL 404)	3		
Electives	17	2 17	3 16
Senior			
Mathematical Programming (CSc 419)	3		
Programming Languages (CSc 451)	3		
Computer Programming Systems (CSc 452)	3		
Multi-Programming Systems (CSc 453)		3	
Computer Graphics (CSc 455)			3
Senior Project (CSc 461, 462)	2	2	
Undergraduate Seminar (CSc 463)			2
Growth of American Democracy (Hist 204)		3	
U.S. in World Affairs (Hist 205)			3
Statistical Quality Control (IE 336)			3
General Chemistry (Chem 124)		4	
Electives	5 16	5 17	5 16

Total Units 198
Total Electives 30

Scientific Applications

California State University, Fullerton -- Computer Science Council (interdisciplinary group) Computer Science program.

Required courses:

Lower Division	Units
Math 150A, B - Analytic Geometry and Calculus.....	8
Math 250 - Intermediate Calculus.....	4
Math 281 - Linear Algebra with Differential Eqns.....	3
Engr. 205 - Digital Computation or Q.M. 265 - Computer Methods.....	3
Q.M. 280 - Computer Language Survey.....	3
Total	21

Upper Division	
Q.M. 364 - Computer Logic and Programming.....	3
Q.M. 382 - Information Structures.....	3
Q.M. 485 - Programming Systems.....	3
Engr. 402 - Digital Logic Design.....	3
Engr. 405 - Digital Computer Design & Orientation.....	3
Math 340 - Numerical Analysis.....	3
Math 335 - Mathematical Probability or Engr. 423 - Engineering Probability & Statistics...	3
Math 435 - Mathematical Statistics or Q.M. 461 - Advanced Statistics or Math 440 - Advanced Numerical Analysis.....	3
Q.M. 448 - Digital Simulation.....	3
Q.M. 363 - Introduction to Management Science.....	3
Eco. 301 - Economic Principles.....	3
Total	33
Electives.....	15
Total	69

The 15 units of upper division electives are to be selected to comprise a concentration in one of the three areas: engineering, quantitative methods or mathematics.

Total Units 124
Total Electives 70
Restricted electives 15
General education requirements are included in electives

Business Applications

De Anza College, Cupertino -- Business/Data Processing Division, A.A. Degree -- Data Processing Major (Business Option)

MAJOR REQUIREMENTS

Data Processing	3	Computers and Society	4
	1	Computer Prog. for General Education	4
	80	Modern Math for Business & Data Processing	4
	52	Principles of Computer Programming	4
	85	Decision Making With Computers	4
	62 or 90	Business Computer Language (RPG); Assembly Language Coding	4
	60	Principles of Operating Systems	4
	61	Business Computer Language (COBOL)	4
62,91 or	96A,B,C,D	(any 2) Business Computer Language; High Level Language Coding (PL/1); Adv. Progr.	8
	65	Systems Design	4
Business	1A, B, C	Principles of Accounting	12
	96	Business and Industrial Organization	4

Total Units 94
Major Requirements 60
General education and other requirements 34

Upon graduation, many students having taken various courses in the field of computing at California State University, Long Beach, have been employed into the production, service and government sectors. For example, many have joined the aerospace industry, banking, insurance, private business, computer manufacturers, education, and government. Some have joined as programmers, systems programmers, systems analysis, and sales. Others have continued on to graduate school seeking their MBA or Masters or Ph.D. in Computer Science at other schools.

Since I can only speak about Cal-State University, Long Beach, we are only now putting together a Computer and Information Science Department. This interdisciplinary department is being established outside of existing schools and is being planned and staffed by those who are already faculty members in other departments. The Department will eventually provide the service courses as well as provide a major in Datalogy -- "The science of the nature and use of data." Since the School of Engineering already has a successful computer engineering option that naturally specializes in hardware and software, CIS has had close affiliations with that school. In fact, the Electrical Engineering Department Chairman, responsible for the computing engineering option, is a member of the planning team.

There are those of us that strongly feel that students graduating from an institution of higher education with degrees other than from the traditional computer sciences, should have taken courses with some substance in data handling. With computers playing a more prominent role in organizations today in the private, public, and government sectors, students in the behavioral and social sciences, in the biological and physical sciences, library sciences, education and business, etc., should be required to have had courses dealing with computer techniques in their discipline. It is not critical for these students to be proficient in the design of computers and software or experts in programming. But, these individuals should have the applied sciences of their field; i.e., the marketing student should understand trend analysis, data gathering, etc.; the production management student -- linear programming, education -- item analysis, sociology -- regression, etc. Rather than each department duplicating similar course content, we are proposing the Computer and Information Science department draw on the faculty at the University who have the expertise in the subject area. They will conduct the various CIS mini-courses for the campus or CIS and cross-listed with the department.

It is our intention for our graduates to have a better grasp of what they are expected to know by their future employer as well as to be a better equipped and valuable employee.

FOOTNOTES

[1]Comm. of the ACM, Vol. 5, No. 6, June 1972, p. 470-472.

[2]SIGCSE Bulletin, Vol. 4, No. 4, December 1972, p. 2-5.

[3]SIGCSE Bulletin, Vol. 5, No. 1, February 1973, p. 102-105.

WHERE ARE OUR UNIVERSITY COMPUTER SCIENCE GRADUATES GOING?

Bruce H. Barnes
Pennsylvania State University
University Park, Pennsylvania

Introduction

In this discussion we will address our attention to where the graduates of the B.S. and M.S. programs in Computer Science are going and note what implications can be drawn relative to smaller educational organizations. This information was gathered in two surveys of the graduates of these programs.[2,4] A survey of the entrance requirements of Graduate programs in Computer Science was also made.* We will use the graduate and undergraduate Computer Science programs at the Pennsylvania State University as a typical model of a University Computer Science program. While the baccalaureate program differs in detail from Curriculum '68,[3] it agrees in spirit. Likewise, the graduate program is in accordance with the ACM Committee on Computer Science Curriculum report on graduate programs.[6]

Education

Before we discuss the placement of these graduates it would be advisable to look at the education they received. The undergraduate program consists of three major parts: general education, related technical and computer science. Seventy (70) of the one hundred twenty-four (124) credits needed for graduation are in the fields of communications skills, foreign languages, science, social sciences, arts and humanities. These courses are included so that the graduates will be able to understand, live in and enjoy modern society. The related technical courses are in mathematics, statistics and technical electives in an area of computer applications. The computer science section comprises twenty-seven credits and is also subdivided into three areas: required basic, required advanced and elective advanced. The required basic courses are algorithmic processes, assembly language programming, numerical calculation and introduction to the foundations of computer science. Systems programming, data structures and the structure of programming languages are required of all students and form the heart of the program. To round out his computer science education, a student chooses two courses from graph theory, numerical analysis, foundations and logical design.

The masters program has four essential ingredients. The first consists of prerequisite material to prepare the student for graduate level courses and to allow him to make up the equivalent of an undergraduate degree in Computer Science, if he didn't have one. Only a small amount of this material can be used to complete the degree requirements. The core of the program is three courses; Structure of Artificial Languages, Systems Programming, and Information Processing Systems. This portion of the program forms the commonality among Master's graduates in Computer Science from most schools. It insures that the student is a proficient programmer as well as having a knowledge of the fundamental subfields of computer science. The third requirement includes sufficient course work to bring the total credits earned to thirty. It allows the student to broaden in some areas and to delve into others. It may include a minor. The last requirement is that the student produce a written paper on some topic in Computer Science. It may be a credit thesis or a less extensive no-credit paper. The important thing is that the student be able to attack a professional problem and describe his work in an intelligent and educated manner.

Employment

Of the forty baccalaureate graduates responding to the questionnaire, twenty-one or over one half had jobs as programmers and 5 had jobs as systems programmers; almost all of the remainder had positions related to computing. As might be expected, the computer manufacturing firms and software service companies employed a large number: 7 and 4 respectively. But the rest found employment in commercial organizations of many types and varieties such as steel, oil, electronics, utilities, etc., which collectively account for 17 of these people. Finally, education attracted 4 while 5 were in military service and 1 accepted employment with the government. Further insight into the career aspirations of these students can be found in their response to the question "What do you hope to be doing five years from now?" Several responses from women placed primary emphasis on raising a family with either part-time professional employment as a programmer or withdrawal from the job market with hopes of returning at a later date when family conditions permitted full time employment. Excluding this female family-oriented group, the responses to the question were:

---

*The author is indebted to Mr. Gerald L. Engle for gathering this material.

Design of Systems Software and Operating Systems	8		
Managerial Positions in Computing	8		
Return to School	6		
Teaching		5	
Don't know	3		
Same as present but at an advanced level	3		
Unspecified job in computing	2		
Research in computing problems	2		
Systems Analysis	2		
Designing Information Systems		1	
Consulting	1		
Business and Commercial Application	1		

Note that multiple or alternate job objectives were stated by a number of respondents. None, it is interesting to note, wanted to be out of the field of computing and data processing; rather an emphasis on moving ahead in the field either in management of computing or systems programming is strongly evident.

The Master graduates had a similar pattern of employment where the largest class of employers is, predictably, the computer industry itself, followed by educational institutions. The data indicates the following:

Type of Employer	No.	
Computer Manufacturer	19	
Software Firms		10
Manufacturers (not including computer manufacturers)	9	
University or School	17	
Government:		
Military Service		6
Civil Service	11	
Research & Development Firms	7	
Service Industries (not including software firms)	5	
Banking and Finance		4
Misc.		4
Total	92	

Job titles further reveal the use to which these people are putting their education. The largest group are employed as various types of programmers, with about an equal number with staff, analyst and instructor titles. The specific data is:

Job Title (General)	No.	
Programmer	23	
Staff Member		14
Analyst		13
Instructor	12	
Engineer		7
Research Assistant		4
Misc.		19
Total	92	

From this we can see that most of the graduates were using the skills they were taught primarily in the computer industry or the educational world.

The Master's graduates also had aspiration of moving into managerial positions, but education and high level technical work were also a major part of the career goals as shown by the following table.

Objective	No.	
Management position		26
Teaching position		11
Return to school	6	
Research position		6
Systems programming		5
Consulting position		4
Own business		3
Other (misc.)		14
No answer		9

Recommendations

What implications does this have for the smaller college? I think that there are at least two significant implications. Few small schools will be able to offer a Computer Science program that is technically as comprehensive as that of the larger universities. While some studies[5] indicate that the shortage of B.S. Computer Science graduates will continue for a few years and, consequently, non-computer science students will be able to get some positions which would normally go to more technically trained students, they will not be able to compete adequately. Thus, the smaller school should try to place their computer oriented students into different positions and career paths than the B.S. Computer Science graduates typically enter. One such area is applications programming.* To do an excellent job in this endeavor it requires a solid knowledge of the application area, a basic knowledge of science and mathematics, and programming expertise. This could be accomplished by a minor in Computer Science, similar to the program recommended by Austing and Engel.[1] While there will be many opportunities in Computer Science, there will be even more in the traditional careers for professionals with some computer expertise. The key to any undergraduate program is to train the student well enough to success at his first job and educate him well enough to be successful at the position he will have twenty years after graduation.

Some graduates of the smaller colleges will desire a career in Computer Science per se. They could achieve this by taking a computer related position and through experience, on-the-job training or continuing education acquire the necessary computer science knowledge or, preferably, he could go to graduate school for a master's degree. As the surveys show, the Master's Degree in Computer Science provides for a variety of entry positions and a greater diversity of career options than the B.S. program. Consequently the small colleges should provide an adequate education to equip their graduates to meet the entrance requirements and succeed once admitted.

The entrance requirements of fourteen Graduate programs in Computer Science** were surveyed. While all schools provided for students to enter without the stated requirements, they felt that entering students should present the following background before entering the program. All fourteen Universities required a B.S. degree in Mathematics, Science, Engineering or similar field. Consequently, they all required math through calculus and all but one required linear algebra. I was pleasantly surprised to note that nine schools required at least one course in probability and/or statistics. One half of the schools required a course dealing with discrete mathematics. I expect that this number will increase in the coming years because, discrete math is the language of computer-related applied mathematics much as calculus is the language of continuous applied mathematics. While only two schools preferred that entering students have a bachelors degree in Computer Science, all required some Computer Science beyond the beginning programming courses. In most cases at least four courses are required. The program recommended by Austing and Engel is probably adequate, if supplemented with the necessary mathematics.

Summary

The data concerning careers for both B.S. and M.S. students in Computer Science indicate that the graduates from smaller schools will not be adequately prepared for a career as a Computer Science professional, especially in the area of systems programming, and the smaller schools would be better off preparing their students for computer related positions in other areas. They should also prepare students for entry into Computer Science careers through graduate education. Each of these requires course offerings beyond the first course. The equivalent of a minor of about 5 courses should be considered a minimum. Student projects either through the computation center or through the academic program is an excellent means for supplementing the students education in this area. I would encourage all schools to use the informal means of education to give their students more programming experience and expertise. The lack of sufficient budget for a large computing system or sufficient enrollments to justify a large faculty in Computer Science does not mean that the smaller schools are to be excluded from Computer Science Education. Through planning, knowledge and enthusiasm these schools can prepare their students for the exciting world of computing.

---

*The area of commercial data processing is also an excellent alternative field. But that topic is beyond the scope of this discussion.

**These schools were Buffalo, Chicago, Colorado, Georgia Tech., Illinois, Ohio State, Pittsburgh, Purdue, Rutgers, Southern California, Stanford, Texas and Toronto.

FOOTNOTES

[1]Austing, Richard H., and Engel, Gerald L., "A Computer Science Course Program for Small Colleges", Communications of the ACM, Vol. 16, No. 3, (Mar. 1973), 139-147.

[2]Barnes, Bruce H. and Gotterer, Malcolm H., "Attributes of Computer Professionals", Proceedings of the 9th Annual Computer Personal Research Conference, SIGCPR of the ACM, 1971.

[3]Curriculum Committee on Computer Science (C³S), "Curriculum '68, Recommendation for Academic Programs in Computer Science", Communications of the ACM, Vol. 11, No. 3, (Mar. 1968), 151-197.

[4]Gotterer, Malcolm H., and Barnes, Bruce H., "The Computer Science M.S. Graduate", Proceedings of the Third SIGCSE Symposium on Computer Science Education, SIGCSE Bulletin, Vol. 5, No. 1, (Feb. 1973), 106-109.

[5]Hamblen, John W., Computer Manpower - Supply and Demand - by Staten. Information Systems Consultants, St. James, Missouri, 1973.

[6]Melksanoff, M.A., "An M.S. Program in Computer Science", Proceedings of the Third SIGCSE Symposium on Computer Science Education, SIGCSE Bulletin, Vol. 5, No. 1, (Feb. 1973), 77082.

SECTION III

WHERE ARE WE GOING ACADEMICALLY?

COMPUTING AT THE EVERGREEN STATE COLLEGE

Fred H. Young
Evergreen State College
Olympia, Washington

The Evergreen State College is a new state college in Olympia, Washington, that has just completed its second year of operation. It has a student body of approximately 2000 at the present time and a few more than 100 faculty members. Our computer operation is based completely on the library model. Every student, faculty member, and staff member may be assigned a number at his or her request that gives him free access to the computer. There is no restriction on time used by an individual. Each person has a limited amount of file space automatically assigned to him for storing programs and data files. If someone needs more space, he requests it from a secretary on the computer staff. So far, no one has been refused, but it is possible that in time space may be limited.

The college's computer is a high speed version of the Hewlett-Packard 2000-C capable of driving up to 32 terminals. Commonly known as "Hewpy," it has a core memory of 40,000 16-bit words, 32K in the CPU and 8K in the front end processor. We have a 1M character fast disk, and our 4.8M character disk packs have just been replaced by a 23M character disk. Our terminals include 19 teletypes (14 hardwired and 5 dial-up), a Portacom, 4 Teleray CRT terminals, an X-Y plotter, and a T4013 graphics terminal. Another terminal on the order of a Hazeltine 2000 scope with printer is out for bids. The plotter and Teleray terminals operate at 30 cps, the T4013 at 240 cps, and the TTY's at 10 cps.

The computer is dedicated to a single time-shared language, an extended BASIC. There are no credit-bearing courses in programming, but the computer center staff frequently gives short workshops in BASIC. Most students, however, are attracted to the computer by an extensive list of games, some developed at Evergreen, some elsewhere.

In addition to Hewpy, Evergreen has an RJE consisting of a card reader, punch, and line printer connected by leased line to an IBM 360/67 at Washington State University. All administrative programs and files are kept on the 360. The RJE permits students free access, in batch mode, to a computer that can process programs in FORTRAN, LISP, SNOBOL, PL/1, COBOL, WATFIV, and an assembler. Most students learn languages as they need them but a workshop in FORTRAN was given last year. We have found, however, that BASIC serves the needs of the students and faculty so well, and is so easy to use, that only a few students are using the RJE.

With no computer courses but with free and essentially unlimited access to a computer, what uses are made by members of the academic community? Much of it is of the expected sort. Nearly all students studying mathematics are expected to use the computer as a calculating tool for such things as numerical integration, finding zeros of functions, making linear transformations, and so forth. Statistical analysis of data, whether scientific or sociological, represents another common use. But since Evergreen's academic program is quite different from that found at most institutions, it is not surprising that the greatest use of the computer is nonnumerical in character. A year ago I conducted a one-quarter program called "Puzzles, Games, and Problem Solving." The students studied Boolean algebra, logic, and problem solving techniques of many kinds. Each student studied intensively the strategy of some known game and invented a game of his own, complete with an analysis of the strategy for playing it. All the students used the computer extensively, and some invented games to play on it. One pair of students came up with a version of SEAWAR, played at two terminals, that is one of the best versions I have seen. They patented it and sold boards for it in our bookstore. Three students programmed versions of Conway's LIFE game, but they were agonizing to run on our 10 cps teletypes, the only kind we had available at that time. One student programmed the sliding block puzzle for a 3 x 3 board. His first attempt, a brute force search of the complete tree of the game, took hours to run. He then rewrote the program to use a polynomial to evaluate board position. This version would solve the puzzle in about 15 seconds. It was amazingly efficient. Another student developed a new kind of condensed tree graph for a couple of nim-like number games. These graphs could then be translated almost directly into computer programs. They now take on all comers at Evergreen. One of the better students wrote a program that constructs extraordinarily complex mazes on the X-Y plotter. A companion program solves such mazes. It is fun to watch it work. One student got interested in the general field of artificial intelligence and programmed a "beast" in a room with objects that possess various degrees of warmth or coldness, hardness or softness. The beast likes warm, soft things and dislikes cold, hard things, but it has an overriding curiosity that causes it reluctantly to leave the things it likes and, avoiding things it remembers as unpleasant, to explore the room for any new objects that may have been introduced. This project was explored for a year and has now been written up and submitted for publication. The student is now working on programs to write music.

I must not forget an unusual poker playing program written by a student. He programmed four distinct personalities (a cautious player, a bold one, etc.) who played against each other and modified their strategies according to whether they won or lost. After they had played a great number of hands, it was found that their strategies converged. The final result is a challenge to the academic community. After all, Hewpy doesn't really collect its winnings.

A staff member has written a hilarious program to simulate a psychiatrist. "Psycho" gets a good play by students. He has also become involved in computer art and has turned out a number of fascinating constructs.

A faculty member with a background in English has made enormous strides in constructing a program to help students learn expository writing. At present one subprogram, called PARSE, is operative. It analyzes such grammatical features as sentence structure and length, paragraph construction, use of modifiers of various types, and conjunctions. The output is a statistical analysis of the student's writing style as compared with that of other students and with professional writing of similar type. The faculty member has recently been funded and will start in September to develop a full-blown CAI program for improving writing skills.

Evergreen has also received a COSIP grant to develop autotutorial materials for student use. Although other devices will also be used, a major effort will be to construct CAI materials for use in the sciences. This project will be in full swing this summer. Several students have already written CAI programs that are being used in a number of ways. Some of these are in statistics, but one of the best is in the fundamentals of programming in BASIC.

Another common use of Hewpy is simulation. One extensive project involved the simulation of tidal action and the mixing of water in a bay in Puget Sound. This program was quite successful and led to some management decisions involving construction, drainage, and sewage disposal around the bay. It may have saved a valuable oyster bed for the Indian community on the bay. One student, interested in book publishing, has simulated an entire publishing house including ware-housing, distribution, labor, markets, and management. It is really an ambitious project. At the present time the student is trying to find a way to include such imponderable factors as worker satisfaction in his model. As you might expect, he is having trouble.

Not all projects have met with success. One faculty member wrote a program for use in first-year coordinated studies that are heavily dependent upon seminars about books the students have read. The program permits faculty with little knowledge of computers to introduce material each week to provide help to students in the most difficult areas. A student can go to a terminal, select the topic that is bothering him, and get a few paragraphs of help. If the student feels that the answers he receives are inadequate, the computer schedules a conference with an instructor. Unfortunately, it demanded more time from the faculty than was available. It was a splendid program. If we ever get our faculty workload down to manageable size, perhaps it will be revived.

This has been a somewhat rambling description of some of the many uses of the computer at Evergreen State College. We are still having difficulty getting some of the faculty to use it. A few, of course, are ideologically opposed to computers, and we must wait until their students force them to reconsider their prejudices. The large majority of students and faculty are delighted with the service Hewpy provides. Being dedicated to a single language, it wastes very little time in overhead. It is virtually impossible to bog it down. Response time is always very short. BASIC is an easy language to learn, and it has eliminated the necessity for extensive programming courses. At the same time, the versatility of the language and the complexity of the programs written by relatively inexperienced students have surprised us. I have been at colleges with larger and more expensive hardware, but never before have I seen such satisfaction with a computer center. Our feeling is that when the time comes that we need to expand, we shall not get a larger computer but will instead get another computer of the same size, perhaps dedicated to another language. Whatever it is, the language will be interactive, and we shall maintain the library concept.

WHERE ARE WE GOING ACADEMICALLY?

Margaret E. Dexter
Augusta College
Augusta, Georgia

In the past ten years or so we have seen Computer Science develop from an occasional course offered at a few schools to an academic discipline offered at most if not all of the larger colleges and universities. Now many smaller and medium sized schools are beginning to offer computer science courses. Most of us have at least one course and realize that we need additional coursework in the area. Those of us who prepare students to transfer to the universities find that our students need some preparation in computer science. Students going on to graduate school in other fields are frequently expected to know how to use a computer, and students entering graduate computer science programs need several courses. Students entering the job market after either two or four years may need some computer background to compete with students from larger schools. But we have neither the faculty nor the computing facilities to offer extensive programs. So where are we going?

First, let's recognize that we do have some advantages over the larger schools. We are typically more concerned with teaching than with research. We usually have smaller classes and increased opportunity to know our students and work with them individually. Although we typically have a teaching load of 12-15 hours we may actually have fewer students than a professor in a university with half as many hours of lecture per week. Perhaps we should ask ourselves how we can use the computer to aid us in our primary function of instructing students as well as whether or not we can or should teach Computer Science. So let's briefly consider instructional uses of a computer.

Instructional uses of computer fall into three general categories: problem-solving, educational games, and tutorial applications. The first instructional use of a computer on your campus probably was or will be -- a problem solving application. And well chosen problem-solving applications can significantly affect instruction. Whether the student programs the problem himself or uses canned programs of some sort, he can be expected to solve problems which would be prohibitive without a computer because of the arithmetic drudgery involved. Hopefully this will permit him to understand and learn concepts because he is freed of the details of computation.

Many faculty members may feel that using a computer is not practical in their courses. They can't finish the book now, and how on earth can they add computer programming and get anything done? The idea is not to pile computer programming on top of the other material in the course, but rather to incorporate the use of the computer into the curriculum in such a way that it facilitates learning other material. We need to keep in mind the distinction between teaching the students about the computer and using the computer to teach other topics. As textbooks utilizing the computer become available this will be easier for us to accomplish. Although changes in curriculum come about slowly, the use of the computer in other disciplines can be expected to have an impact upon the curriculum of those disciplines.

Educational games or simulations are also being used in many areas. Here again the objective is not to learn to use the computer but to use the computer as a tool to learn something else. Rather than presenting the student with a mathematical model of a system in algebraic form, the model can be implemented as a computer program. The student can vary parameters in the model and have the computer demonstrate the effect, frequently graphically. We may also program models far too complex to present mathematically. Although we should remind our students -- and ourselves -- that we are experimenting with a model and that the validity of the results depends upon the validity of the model, instructional applications based on simulation appear very useful and it is reasonable to expect more of these applications in the future.

Finally, the computer can be used for tutorial applications. We are all familiar with programmed instruction or PI texts, and a computer can be programmed to present information to a student, ask the student a question, and then use the student's response to determine the next computer action. For a discussion of programmed instruction see Meadow.[2] It is not clear that there is any advantage to using a computer rather than a book for this purpose. In a paper discussing the directions for research and development of computers in the instructional process published about a year ago, Zinn stated that this form of CAI has earned a reputation as "an unimaginative, costly, page-turning teaching machine".[3]

The emergence of author languages such as Coursewriter and others has presented instructors with an easy way to present materials in an unimaginative way. Although some good courseware has been developed this way, it is extremely difficult and time-consuming to produce good materials and easy to produce bad materials, with the result that a lot of bad materials have been produced.

A major disadvantage of this approach is that the instructor determines what material is to be presented and the student cannot control his own learning. This means that the instructor who prepares the materials must spend a great amount of effort predicting what the student will do and

prescribing the computer action. And the student cannot control what he is to learn or ask questions about material he does not understand.

In the same paper cited above, Zinn notes that the current trend in CAI is toward systems in which the student can control the dialogue. At that time several systems were in the experimental stage but none ready for practical application, largely because of the difficulty of constructing the data base for a conversational application. As a by-product of a study of the question of whether the student or the computer should control the information flow in a tutorial dialogue, a conversational CAI system was recently developed which, based on initial data, appears feasible for practical application.[1] So we may expect a conversational form of CAI in which students may ask questions and browse through instructional material to be available in the near future.

In conclusion, I would like to make a few predictions. I predict that a computing facility will become as much a necessary academic resource as a library and will be required for accreditation. (After this presentation, Joyce Little, Community College of Baltimore, told me that in at least one instance this prediction has already come true.) I also predict that we will go through a phase in which we use the computer for the sake of using the computer and will discover some instructional applications that are useful and discard others. The computer as an instructional tool will be overused and misused until we discover where it is appropriate, and (hopefully!) it will finally take a place as a useful instructional tool used in conjunction with other forms of instruction.

FOOTNOTES

[1]Dexter, Margaret Elizabeth. A Study of Information Control in Computer-Aided Instruction. Ph.D. Dissertation, School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Georgia, 1972.

[2]Meadow, Charles T. Man-Machine Communication. Wiley - Interscience, New York, 1970.

[3]Zinn, Karl L. "Computers in the Instructional Process: Directions for Research and Development." Comm. ACM, 15, 7 (July 1972) pp. 648-651.

FUTURE PROSPECTS FOR COMPUTER SCIENCE

Richard H. Austing
University of Maryland
College Park, Maryland

There are numerous approaches one could take to address the question "Where are we going academically in computer science?". Certainly, all of the other questions that are panel titles at this symposium relate to the question addressed by this panel. The current status of computer science, the kinds of students we produce and the varieties of software and hardware available for use are all vitally important aspects of an analysis of where we are going academically. The discussions on each question provide a necessary context for discussions on all the others.

Because the future of computer science, like any other future, is grounded in the past, I would like to consider the growth of computer science as a discipline during the past 10 years or so. This brief review is intended to be illustrative rather than exhaustive. Only a few published papers and reports will be cited.

In many ways, we have come a long way since 1963 when Saul Gorn's article "The Computer and Information Science: A New Basic Discipline" was published in the SIAM Review[1] and when the first major panel discussion on computer science courses took place that summer at the ACM meeting in Denver. Unfortunately, there are still a number of areas in which we almost haven't even gotten off the ground.

The ACM panel consisted of the following topics and speakers:

1. Programming of Digital Computers (A.J. Perlis)
2. On Introducing Digital Computers (B.W. Arden)
3. An Undergraduate Curriculum in Numerical Analysis (G.E. Forsythe)
4. Logic for the Computer Sciences (R. Korfhage)
5. Mechanical Languages: A Course Specification (S. Gorn)
6. The Place of Logical Design and Switching Theory in the Computer Curriculum (D.E. Muller)

A report of the panel discussion appeared in CACM in April, 1964. This discussion was instrumental in the establishment of ACM's Curriculum Committee on Computer Science (C^3S). This committee, supported by NSF funding, produced two reports in 1965 and 1968 respectively:

a) "An Undergraduate Program in Computer Science - Preliminary Recommendations"[2]
and b) "Curriculum 68: Recommendations for Academic Programs in Computer Science"[3]

I won't list the 16 course titles in the 1965 report and the 22 titles in the 1968 report, but those of you familiar with "Curriculum 68" can easily see the growth in the number of topics considered to be in computer science during the period from 1963 to 1968. C^3S still exists and has every intention of producing an updated report on computer science curriculum, but the termination of funding several years ago has considerably slowed the progress of the Committee.

As we all know, the impact of "Curriculum 68" was tremendous. The number of degree programs in computer science has mushroomed; textbook writing activity has been brisk; attempts at teaching various courses has resulted in the modification of course content, in the reduction of course levels, and in the establishment of new areas and topics. Because of the immense scope of "Curriculum 68" only those institutions with substantial resources could hope to implement a sizeable percentage of its recommendations. This has led to developments within computer science that have been both beneficial and detrimental.

The larger and/or more prestigious universities have acquired faculty and have established computer science departments offering degree programs based, more or less, on the recommendations in "Curriculum 68." In many instances, graduate degree programs were established first. The necessary attention to research and theoretic development required to maintain graduate programs has contributed greatly to the growth and prosperity of computer science as an academic discipline but it has also tended to give computer science an aura of being a "pure" (as opposed to "applied") field. Whether or not computer science will continue in this direction depends on many factors and I do not intend to get into all of the ramifications of the "pure" vs. "applied" debate nor into definitions or descriptions of the terms themselves. I do feel, however, that it is important to cite the situation and to raise the following points:

a) the extent to which "pure" and "applied" are mixed will greatly influence the degree of future success of computer science;
b) computer scientists should be aware of the situation within mathematics and should learn from it;
c) the capabilities, uses, and impact of computers are essential ingredients in the education of a computer scientist; and

d) the continued development of undergraduate programs is vital for the supply of responsible and knowledgeable computer oriented people to the work force and, to a lesser degree (because a smaller number of students is involved), for graduate computer science programs.

I have deliberately not discussed specific topics or areas within computer science nor what may or may not be appropriate to the discipline. I have avoided doing so because any such discussion involves a consideration of local conditions such as background and interests of faculty members, strengths and weaknesses of other departments, and historical development of computer oriented course work. These problems are too diverse to treat in this paper.

Instead, I want to focus on the appropriateness of computer science as a subject within a liberal arts environment. To a great extent, the future of computer sicnece depends on its inclusion as a respectable major field of study in liberal arts institutions. Clearly, this involves a commitment to acquiring and maintaining resources, both faculty and equipment, that such institutions are generally not ready or able to make, even if sufficient resources were presently available. I cannot predict how and when these problems will be overcome, but I can hope that means will be sought. In the interim, however long that may be, development of computer science courses and curricula within liberal arts college should be pursued. Two papers offering guidelines for development are included in the references.[4,5]

Some very practical arguments can be made for any institution, no matter what its size, to offer computer science courses and degree programs. I think they apply particularly to small colleges. For example, admission requirements for graduate degree programs in computer science have been increasing steadily and will, probably within the next several years, include a bachelor's degree in computer sience; already, three or four upper division computer science courses are required. The majority of students, however, will take computer science courses to become more employable or to be better equipped to handle themselves depending on how strongly an individual institution feels about vocational training as an appropriate goal of a liberal arts curriculum. Another argument on the practical side is that more and more high school students are becoming exposed to computing. They will look for more of it in any college or university to which they apply.

Irrespective of these arguments, computer science is an appropriate field of study for a liberal arts student. Any graduate of a liberal arts institution to be considered "educated" surely must have some knowledge about computers. The amount and type of knowledge is arguable but not the fact of having it. A liberal arts graduate with no exposure to computers or computer science is unable to communicate with a large segment of those people within the graduate's sphere on even the most basic matters related to computers and their impact. Because the computer is a useful tool in at least some aspect of all disciplines, the unexposed major of any field is at a disadvantage.

More importantly, the problem solving method inherent in computer science could be the most useful idea given to liberal arts students. The processes involved in breaking a large problem into a set of smaller ones, constructing a model for each smaller problem, devising an algorithm for its solution, testing it, reformulating the problem if necessary, etc. are applicable to any discipline and undoubtedly are a great aid in helping students to think, plan, create and organize.

A student can be introduced to this process in one or two basic programming courses. Additional (usually upper division level) computer science courses will develop the process as well as provide a body of knowledge not included in other fields. Some of the areas appropriate to upper division computer science work could be labeled data structures (or information processing), languages (usually with emphasis on structure and properties rather than on developing programming skills), numerical analysis, simulation, systems (including computer organization), and theory of computing. Good textbooks are becoming available in these areas so that it is possible to teach concepts in these areas (in contrast, for example, to teaching only the details of a specific piece of equipment). "Concepts" is the keyword here, because a student with a command of concepts will have the flexibility to adjust to a variety of environments whether on the job or in graduate school.

I am not concerned at this point in time with the question of what kinds of jobs a computer science person will be qualified for. Current graduates with computer science backgrounds are getting jobs. Furthermore, as more and more computer science majors graduate, positions suitable to their backgrounds will become better defined.

Both in looking into the past and in looking to the future, I see a number of needs and problems that continue to plague us and which bear mentioning: a) Teacher training, both in-service and for secondary school teachers, must be increased; b) Trial and error course and curriculum development must be brought under better control; c) Institutions with resources must do more reaching out to help those with little or no resources; d) Better articulation must be achieved among personnel in 2 year, 4 year, and graduate programs; e) More attention must be given to innovative pedagogical methods; f) Service courses must be improved so that they better suit student and department needs; and g) Continuing education must be implemented on a greater scale to assist computer professionals in keeping up with the field.

In summary, I feel that the future of computer science depends on how at least the following questions are answered:

1. What kind of mix of "pure" and "applied" subject matter will be maintained?
2. How successful will small (liberal arts) colleges be in implementing computer science degree programs?
3. How rapidly and well will the needs identified in the preceding paragraph be met?

FOOTNOTES

[1]Gorn, S. "The computer and information sciences: a new basic discipline," SIAM Review, 5, 2, (1963), pp. 150-155.

[2]ACM, "An undergraduate program in computer science - preliminary recommendations," CACM, 8, 9, (1965), pp. 543-552.

[3]ACM, "Curriculum 68," CACM, 11, 3, (1968), pp. 151-197.

[4]Austing, R.H. and Engel, G.L. "A computer science course program for small colleges," CACM, 16, 3, (1973), pp. 139-147.

[5]La France, J. and Roth, R.W. "Computer science for liberal arts colleges: a report of a workshop held at Wheaton College, Wheaton, Illinois, July 12-14, 1972," SIGCSE Bulletin, 5, 1, (1973), pp. 70-76.

SECTION IV

WHERE ARE WE GOING TECHNICALLY?

COMPUTER-BASED LEARNING IN 1980: WHAT WILL IT BE LIKE?

E.D. McWilliams
National Science Foundation
Washington, D.C.

Anyone who has experienced - as many of us have - the disparity between promise and performance for computer (and other) technology should be rather cautious in predicting future capabilities. I was asked, however, by the CCUC committee, to be "provocative," and in that spirit, I am prepared to predict that by 1980, CBL (Computer-Based Learning) will be characterized by the availability of a wide range of low cost, high performance educational computing systems, capable of providing highly effective instruction in most disciplines and educational levels. Furthermore, CBL will be acknowledged, finally, as the most important educational innovation since printing, and most educational institutions will either be using it routinely or will be making plans to do so.

I hope that at least some of you consider this to be a "provocative" statement. I believe it to be a reasonable one, however, and one upon which your institution should be informed, especially since, in my opinion, only a severe depression or other national emergency can delay this prediction by more than a few years.

Notice that I've used the euphemism "CBL" instead of "CAI" or other more restrictive term. Within CBL, I embrace all present forms of educational computing, and then some. Our educational system is characterized - fortunately, in my opinion - by diversity, and so it will be with CBL. Such systems will differ markedly in matters of:

- scale (ranging from dedicated, self contained, single terminal systems to huge instructional utilities serving thousands of terminals scattered all over the country).
- function (CAT, CMI, CAI, "dual and solo mode" problem solving, simulation, and "browsing").
- facility (libraries of software or courseware programs, programming or authoring services, information storage and retrieval services, graphical input/output, natural language input/output, and so forth).

(We shall not want for variety - that is the surest prediction of all!)

Before explaining the basis for this prediction, let me explain the program within the Foundation that I administer, since it provides the perspective for my opinions. My program - Technology and Systems - is one of 3 within NSF concerned with "Technological Innovations in Education". I'm responsible for ensuring the existence of better (and better) hardware, software, and "courseware" (instructional computer programs), by supporting research, development, and evaluation of promising techniques and systems. This program is presently supporting a number of interesting projects, notably the development and field test of the PLATO and TICCIT systems of CAI. [I'll describe these systems only briefly here, in order to develop other bases for my prediction. I did bring along a few reports, and I'm prepared for questions later.]

PLATO (IV) is intended to be an instructional utility, capable of providing CBL from a large library of courseware, to thousands of widely scattered students simultaneously. TICCIT is intended to provide highly structured CAI to roughly a hundred students simultaneously, in community college English and mathematics. (One TICCIT system is designed to serve a single community college.)

Each of these systems offers several interesting and seemingly valuable instructional features, through the use of significantly different technologies and strategies. For example, PLATO's (plasma panel) console provides quite useful, high resolution graphics, color slides, touch input, and audio messages to remote locations using standard telephone lines. TICCIT's "Digicolor" television console will provide high resolution - several times better than standard television - display of color characters or piecewise graphics, plus videotape and audio capability, at a very modest cost (under $2,000 capital cost per console). Furthermore each system is designed with efficiency (as well as facility) in mind; each is intended to deliver CAI at a cost - exclusive of the cost for developing the courseware - of $1.00 or less per student-contact-hour.

The strategies for achieving their educational goals are markedly different as well. PLATO is designed to enable a faculty member to prepare his own courseware, according to whatever educational strategy he practices. TICCIT's entire design - right down to the "Learner Control Keyboard" and other hardware - is based upon a rather simple, general model of instruction in which the courseware is prepared by teams of "courseware craftsmen", and delivered and executed as data.

Based upon their early promise, the Foundation began two years ago to organize a major field test of these two systems, primarily at the community college level. Progress for the field tests has been good - we seem likely to miss our original starting dates by only 6 months. (The original date was September 1973 for PLATO, and February 1974 for TICCIT.)

I believe that each of these systems will meet its design objectives - namely to provide effective CAI at a cost of $1.00 per hour - before 1980. By themselves, therefore, they will indice considerable change in the use of CBL, since the cost for human instruction is already several times that figure. There are many other encouraging developments too, however, and I see strong promise from several that can be grouped loosely together under the heading of "machine intelligence".

For example, computers are already able to communicate with humans in natural - though still restricted - language, by recognizing and reproducing human speech, with ever-increasing accuracy and speed. Natural language communication should be quite far advanced by 1980, and could prove quite useful for CBL.

Furthermore, computer programs are increasingly able to understand the substance of what they teach. For example, I recently inspected a program capable of instructing in integral (as opposed to numerical) calculus. It understood the various methods of integration - by parts, trigonometric substitution, etc. - and where each method applied. It was therefore able to solve its own (randomly generated) problems, or others posed by the student. With this rather deep understanding, it was able to provide the student with specific advice - e.g., "perhaps a second substitution of variables would help" - when requested. (This program had an additional nice wrinkle; it was able to learn from the student. If the student's strategy for a particular problem proved more efficient than the programmed strategy, it was adopted by the program, and would be used for such problems in the future.)

There are promising developments in programming languages as well, such as just reported to us by Dr. Mills. I'd like to mention another, that I find especially exciting.

All programming languages with which I am familiar are sequential, like the machines upon which they run. In the real world, of course, events proceed in parallel, and often independent (asynchronous) of one another. Computer simulations of real world events using standard sequential programming languages have therefore proved to be quite complex, as anybody who has attempted one soon discovers.

At the same time, simulations of real world phenomena are widely recognized as potentially valuable learning aids. Realistic simulations and graphical displays of ecological and other systems are already popular. The programming required to build (rather than merely use) a simulation prevents many interested faculty members - and even some students - from attempting to write one. This seems unfortunate, since the insight into the process of the simulation will surely be deeper when the simulation is constructed, rather than merely exercised.

One researcher has developed a programming language based upon this notion of parallel processes (and other promising concepts), which enables one to program the processes in the manner that he thinks of them. For a space war simulation, for example, one describes the characteristics of one ship (speed, acceleration, heading, field of vision and fire, etc.), then kicks it off and concentrates upon launching the second ship, or third, or whatever. Each ship proceeds on its programmed course, with coordination - the essence of simulation - between ships and earth provided automatically at the systems level. I saw a listing for such a program, written in this language (by a student, of course); it consisted of a half page of source code! (Although probably significant in itself, the length of the program is not the advantage being sought, but rather a much better match between the natural thought process and the resulting program code, to reduce unnecessary thought processing and memory load. This is of course the same objective being pursued by the advocates of "Structured Programming", reported to us just moments ago by Dr. Mills.)

I must mention too the promising attempt to stimulate learning in young children through the use of physical models or other devices, whose principles of operation can be exercised by computer programs written by the children themselves. For example, a computer controlled, 3-wheeled cart can be moved about by program control, and leave an ink tracing of its circuit around the room, to illustrate the principles of plane geometry. Similarly, the principles of structural biology can be exercised through the use of a computer-controlled physical model of an animal, with moveable parts, "muscles", and "nerve networks".

Many other promising technological developments could be cited, but let me mention an equally important phenomenon, namely the concurrent development of the human resources required to exploit this promise. People are becoming very experienced - even sophisticated - in the application of technology to education, and this ability seems to be growing exponentially. Computing is infiltrating departments previously thought to be impervious, for example in the arts and humanities. At this conference alone I've heard reports of the use of APL - a sophisticated mathematical programming language - by university English majors (not to mention third grade elementary students). Furthermore, I perceive some understanding not only of the application of computing, but the (deeper) process as well, which seems likely to be of fundamental importance for the reform of curriculum and for the provision of new educational services. As an example of the latter, I can report that my program is presently supporting the development and field test of a computer-based system of career guidance and planning that seems certain to provide to college students, well before 1980, insight into their own values and opportunities, and skill in gathering information and exercising choices - skills of acknowledged value that are virtually nonexistent today. The interaction between computing-wise students and computing-wise faculty seems to be very

promising indeed, especially when supported by theories and conclusions from computing-wise research.

One might conclude from all of this that the millenium is close at hand, except for cost, but here, too, the signs are quite encouraging. The cost for most basic electronic components has gone down steadily by a factor of 10 in the last 4 or 5 years, and is expected to continue this trend for the next 5 years. (For example, the cost for computer memory, disk memory, and integrated circuits has been so affected. Rumor has it that IBM is simply crushing returned 2314 disk units in a hydraulic press like junked cars, as the most effective form of salvage. The 2314 was the industry standard only 2 years ago.)

Concurrently, computer performance and reliability have improved dramatically, through the use of medium and large scale integrated circuits for non-destructive-readout memory and central processors. This has not only improved performance by reducing the distance required for signal travel, but reduced the physical size of the equipment as well. Computing power that required an entire room 10 years ago could be available in notebook size by 1980, and research is well along in one commercial laboratory to produce just that - a protable, battery powered computer system, with console, CPU, main and secondary memory, contained in a notebook. (If this doesn't seem ambitious enough, consider that the head of the project believes that such a "personal computer" could be produced for less than the present cost of a graphics console alone!)

Having thus explained the basis for my prediction, let me hasten to state what I am <u>not</u> predicting. I don't think for a minute that even getting to 1980 will be a breeze, or that we will in fact find everything rosy when we get there. Getting access to the technology won't be easy; integrating it into the curriculum will be even harder. A lot of people will be working very hard to do so.

Furthermore, I cannot reassure you that this technology won't replace some faculty. With the cost for human instruction going up and the cost for CBL coming down, some replacement seems inevitable. The way in which this advantage is converted will be a local decision, determined by those local factors - such as growth - that dominate. There is no reason to believe, however, that 1980 (or, say, 1984) will necessarily be Orwellian - with the technology allowing the government to dominate <u>us</u> - although recent events should encourage us to remain constantly on our guard.

Some wag recently compared CAI with a waltzing elephant. People, he said, weren't for a minute impressed by the quality of the elephant's waltz, but only that it had the audacity to try! I hope that it is clear from my remarks that I believe such an analogy to be very shortsighted. Within a very few years, that elephant will be doing some very nice steps indeed, and a good many intelligent people will be dancing right along with it. Your institutions, and mine, should be preparing for the occasion.

TRENDS IN THE DEVELOPMENT OF COMPUTER SOFTWARE FOR COLLEGE ADMINISTRATIVE APPLICATIONS

T. Ray Nanney
Furman University
Greenville, South Carolina

What will be the nature of administrative software development for colleges and small universities during the coming decade? In my reading of the computer literature and in conversations with many individuals I have not found any clear-cut agreement about the trend of future developments for administrative software. Consequently, you will be hearing my opinion. Moreover, since my experience has been primarily in small institutions, my remarks will be about them.

Before discussing more technical matters, there are several general questions that each one of us should consider.

1. Are we doing the right things with the computer?
2. Is the computer center being managed in the best possible way?
3. Is top administration involved in establishing the computer center policies?
4. Are we insuring the privacy of the computer records for individual students, faculty, and administrators?

Personally, I believe that the success or failure of a particular computer installation depends more upon these factors than upon technical matters.

As a first step in speculating about the future, we all need to ask whether our computer centers are doing the right thing. My impression is that many of us consider the computer to be a super toy to be used in part for our intellectual amusement, and we may even look upon administrative applications as large puzzles to be solved so as to optimize the effeciency of operations within the computer center. This attitude must be changed. This subject is receiving increased attention from the computing community, and many excellent articles regarding it are found in Fred Gruenberger's recent book, Effective Versus Efficient Computing (Prentice-Hall). All computer center directors will find this little book to be extremely helpful and enlightening.

How do we tell whether or not our computer centers are doing the right thing? A reasonable start can be achieved by classifying the output of our programs for administrative offices using the scheme of B. Van Dusseldorp.[1] He divides the information needed for colleges to function into three levels:

1. information for operations -- i.e., for conducting clerical tasks such as payroll, student records, and financial transactions
2. information for control -- i.e., for implementing administrative decisions and policies such as budget control
3. information for management decisions -- i.e., for formulating management decisions and developing policies such as determining tuition rates and deciding when to construct new buildings

Unless your situation is exceptional, your computer center produces little information which is used for control or management decisions. Effective use of our computers will come as we learn how to provide these higher levels of information. Progress is being made in this respect by the Western Interstate Commission on Higher Education (WICHE), by the National Laboratory for Higher Education (NLHE), and by many colleges and universities.

My second general observation is that even small institutions must have computer center directors who have management skills. This is more important than having a knowledge of computers. Most of the college computer center directors that I know were previously either professors, computer programmers, or systems analysts. (This may or may not be a true example of the Peter Principle.) There is no necessary correlation between these occupations, interesting and important as they are, and a knowledge of management techniques. When was the last time you actively sought to improve your management skills by reading management books, discussing a point of management with a knowledgeable colleague, or enrolling in or auditing a management course?

The third general point is that top management must participate in establishing and enforcing policies for the computer center. This is essential but difficult to accomplish. Many top administrators in small institutions have little or no knowledge of the computer or its operation, and they are in awe of those who do. Many have not recognized that they must give just as much attention to the management of the computer center as they do to any other important part of the organization. The result, in some cases, has been an improper abdication of many management functions to computer center directors. Who sets the priorities for the work done by your center -- was it one of the top three administrators in your institution? In your school, does the "squeaky wheel" get computer services first? Do you have a list of guidelines which were established by your superiors?

My final general point is that we must insure the privacy of the computer records of all individuals regardless of the nature of their association with the school. The public is rightfully aggravated with the mistakes caused by improper use of the computer and the resulting inconvenience and trouble. Many are concerned about the possibility of a federal data center. Supreme Court Justice William O. Douglas has stated: "The manner in which government pries into men's lives and fills their personal files with data of this sort makes the arrival of a federal data center a most dangerous event."[2] In designing computerized systems for administrative offices we have an opportunity and an obligation to demonstrate to our students, who, after all, are tomorrow's leaders, that computers need not be feared when they are used properly.

What can we expect technically in the next decade for computer applications for administrative offices? We will surely see (1) significant improvements in our ability to deal with change, (2) new languages having more powerful instructions than those we are presently using, (3) sophisticated programs for administrative applications that can be purchased for a fraction of their development cost, and (4) powerful programming aids that can be executed on small computers.

There are at least four different types of changes with which we must cope in our computer centers: (1) changes resulting from improved technology, (2) changes resulting from a shift in the policies of our institution, (3) changes resulting from the resignation and employment of new administrators, and (4) changes resulting from modifications in the educational system itself. At Furman University we have an IBM 1130 computer, a rather large one as 1130's go, but we are having difficulty doing all the work our administration wants. Moreover, many schools are having similar difficulties and will soon be forced to upgrade their equipment. Within the last year or so, improved technology has made it possible for us to consider a wide range of attachments to the IBM 1130 to improve its performance and also to consider many other computers as replacements for the 1130. To illustrate the complexity of the problem: among the systems that can reasonably be considered are the Burroughs 1700 series, CHI 1130, DSC Meta 4, GA 18/30, Hewlett-Packard 2000 and 3000, Honeywell 115, IBM System 3, 360/22, and 370/115, Logicon 1130, NCR Century 100 and 200, PDP 11/40 and 11/45, XDS Sigma 3 and Sigma 6. One of my greatest fears has been that it would be very difficult to change computers because of program conversion problems. Fortunately, program conversion does not appear to be a significant problem now because the manufacturers are designing their software to minimize conversion difficulties.

Advances in technology will also make it possible for small institutions to consider having on-line administrative systems. At Furman for several years some administrators have wanted such systems, but when they learned of the cost, they backed away. The low cost of the recently announced DEC DATASYSTEM 340, which leases for as little as $1200 per month and which provides many of the on-line features that administrators want, demonstrates clearly that we must soon give attention to on-line systems. How much would it cost your school to convert its computer applications for administrative offices to an on-line operation? It is not too early for us to design our systems with the conversion in mind. We need to know what we will do with our batch-oriented systems when large data bases and time sharing become inexpensive.

To be effective our computer centers must be responsive to sudden changes in policies, administrative personnel, and the educational system. As an example, in 1969 complaints and rumors that Baptist students were being refused admission to Furman, which is Baptist supported, caused the administration to give high priority to developing an admissions data base and suitable admissions reports. Because we had a generalized information retrieval system, it was possible for us to implement the essential parts of the system immediately. It was shown that Baptists were being treated quite fairly, and possible embarrassment and loss of funds were avoided.

Changes in administrative officers of an institution cause complex problems for a computer center. A new administrator may want "minor" changes in the form of various reports, but making these changes may not be minor at all. The management style of a new administrator may require that new reports be developed as quickly as possible for him. How can small computer centers with their limited staffs respond adequately in this situation? My belief is that we must make maximum use of recently developed new concepts of programming techniques to reduce the work of writing new systems. In his 1972 Turing lecture, Professor E.W. Dijkstra predicted: "...well before the seventies run to completion, we shall be able to design and implement the kind of systems that are now straining our programming ability at the expense of only a few percent in man-years of what they cost us now, and besides that, these systems will be virtually free of bugs."[3]

Most college computer center directors have a horror story they can tell about the operation of their centers. My horror story has to do with printing report cards and with changes in the educational system. In 1967 when our first report card program was written, Furman was using a semester system. In the fall of 1968 a three-term system was introduced in which all courses were of equivalent weight (4 semester hours), and progress toward graduation was measured in courses earned. The report card program was rewritten, but we were clever. We assumed in rewriting the program that a monolithic course structure would not last, so we prepared for .5 courses and double courses. Within a year these were introduced, followed soon by pass-fail courses and non-credit courses. In 1971, despite vehement claims by administration and faculty that it would never happen, .75 courses (3 semester hours) were introduced; we discovered that allowance had not been made for this possibility, and two man-months were used rewriting the report card program. Following this change the program was sufficiently general that after trivial changes, it was used by three other colleges. One task of the program is to apply a complex set of rules to determine whether a student

goes on academic probation. Each year during the existence of this version the probation rules have changed and the program was patched. The last patch in the summer of 1972 required nearly a man-month and used a great deal of computer time for testing. Last month, May 1973, the probation rules were completely changed. Unfortunately, the program was written as one large mainline program (31 pages including comments), and it must now be rewritten.

How could all of this work have been avoided? I believe that the answer lies in modular, top-down programming. Structured programming as expounded by Dijkstra results in reliable programs that are easy to modify. Such programs can also be generalized with respect to data bases. Using these techniques, it is my opinion that the report card program could be rewritten so as to apply to almost all institutions. Specialization of the program to a particular institution would require only simple modifications to some routines. New insights into programming techniques are to be expected as we gain experience with Dijkstra's methods.

In addition, advances in techniques for designing software can be expected, and these advances are likely to be useful to those of us who are operating small computers. As an example of improvement in software design, consider the Automatic Engineering Design (AED) system which was designed as a software engineering system. D.T. Ross[4] has stated regarding AED: "It is now possible for management to control the use of resources in software development. From the first prototype on, each stage entails minimum expenditure to achieve a well-defined goal." The AED system includes an integrated, modular library of software components which can be quickly combined to form a working prototype for a desired system. The resulting system can be operated on a trial basis, and suggestions for change can be solicited from the user. The suggestions are then incorporated into the system, and the process is repeated. Proceeding in this iterative manner, the system is developed. When the user is satisfied, the system can be fine-tuned to obtain improved efficiency. The admissions system for Furman was developed in this iterative fashion. Results were available as soon as the data were collected, and the final system was both inexpensive and responsive to the needs of the admissions office.

During the seventies we will also see colleges invest more in programming packages developed by other educational organizations and also by software companies. Why should we continue to redevelop administrative systems at our institutions when the job has been done elsewhere? For example, NLHE has just begun marketing a generalized general ledger system specifically designed for colleges and universities. The system sells for $1000, and NLHE literature states that colleges could not design a similar system for less than $9000. Actually the system cost NLHE several times their indicated minimum for development. In these times of financial difficulty for our educational insitutions, it makes particularly good sense to invest in this and similar software packages.

Colleges and universities are far behind business and industry in the purchase and use of general-purpose programming packages and programming aids. In 1972, at least 28 different file management packages were being marketed and over 1700 installations had been made. All of the packages reduce the effort of developing software by providing generalized file management and information retrieval capabilities. A personal friend of mine, who directs a large business computer center near Furman, told me that more than 60 percent of the programming in his center is done with one of the packages. He claimed a factor of 6 to 1 in reducing programming time and further claimed the resulting programs ran faster than ordinary programs. For small educational institutions the delay in using these programming aids is understandable. How many of us can afford to spend $21,000, the average cost of these systems? As the seventies pass we can expect file management systems and programming aids like metaCOBOL to become less expensive and more of us will have the opportunity to use them.

The anticipated changes in the nature of software for college administrative offices is exciting to me. The intellectual challenge of developing structured program is particularly stimulating. I look forward to the time when we can do our work in a truly effective manner.

[1]R. Van Dusseldorp, "Some Principles for the Development of Management Information Systems," Management Information Systems in High Education: The State of the Act, edited by C.B. Johnson and W.G. Katzenmeyer, Duke University Press, 1969, page 29.

[2]Computerization of Government Files, What Impact on the Individual? Reprinted from UCLA Law Review, Vol. 15, No. 5 (1968), American Bar Foundation, Chicago, Illinois, page 1376.

[3]E.W. Dijkstra, The Humble Programmer, Communications of the ACM, Vol. 15, No. 10, October 1972, page 859.

[4]D.T. Ross, "Fourth-Generation Software: A Building-Block Science Replaces Hand-Crafter Art," Computer Decisions, April 1970, page 32.

SECTION V

ADDITIONAL CONTRIBUTED PAPERS

THE COMPUTER IN TEACHING--TEN WIDELY BELIEVED MYTHS

Alfred M. Bork
University of California
Irvine, California

I will discuss here some aspects of the computer as an educational tool in many disciplines. My teaching field is physics at the university level; however, I think that many of my findings are more broadly applicable.

My plan is to review a number of common myths about the computer in teaching. These myths, while not universally believed, are widely held. Within this framework I will try to indicate where we are and where we may be going with the computer as a teaching tool.

Myth 1: You Must Choose Between Direct and Adjoint Use of the Computer

Literature about the computer in learning has correctly stressed that the computer can be used in two ways. Either students can do their own programming, using the computer as an intellectual tool - sometimes called the adjoint use - or students can interact with teaching programs prepared by others - the direct or mainline use. However, much literature tends to go beyond this, stating or implying that a choice must be made between these two. During the past three years some major developments in educational computing have chosen between the two uses, making it difficult, either because of equipment of socio-political factors, to engage in the other approach. I see absolutely no reason why a teacher should be obliged to make this choice. Satisfactory examples exist of the computer being used both ways, and so neither need be ruled out on philosophical grounds. The same can be said about the many types of dialogs, interactions between teacher and student via the computer. Probably certain types will prove to be efficient for particular subject matter areas, whereas other subject matter areas may require different kinds of dialogs.

Myth 2: You Must Have Massive Equipment to Use the Computer in Education

Some of the more interesting teaching applications have come from schools with minimal computer equipment. Small standalone minis certainly do rule out some of the kinds of things that can be done. Thus, dialogs are not possible on small minis; but many other types of usage are possible. The idea persists that one can start only at the level of huge installations, but innovative teachers have shown it to be wrong many times.

Myth 3: One Language is Much Easier to Learn Than Another

When students write programs for problems in a physics or mathematics course, the questions of which language and how students learn that language become important. Arguments are often based on supposed easy learning of one language or another; thus proponents of BASIC often claim that it is very easy to learn. My own experience indicates that the way the language is taught is much more important than the language itself in determining speed of initial learning. If a reasonable subset is picked, and if reasonable ways are used to introduce the language to students, almost all commonly used languages are relatively easy for beginning students to learn and to use. In my opinion, differences between initial learning ease have been much exaggerated.

Myth 4: Computers Will be Widely Used in Education in Present Organizational Structures of Institutions

New educational developments are often assumed to fit into existing institutional structures. However, it appears to me that the computer is almost certain, in the long run, to revolutionize the organization of schools and universities. The ability to provide learning materials at any time and at any pace, and provide self testing features, the ability to respond individually to students, to have access to large amounts of data, all these imply that the way schools operate are likely to change drastically when computers are widely used in learning.

One interesting view of how this revolution might happen is presented in Education and Ecstacy (George Leonard).

Myth 5: At This Time Your Best Buy for a Terminal is the Model 33 Teletype

I feel strongly that educational users should never buy model 33 teletypes today! This is an old, unpleasant device which types at a slower speed than students read. Its noise level is intolerable, particularly when several are placed in a single room. The argument for the economy of model 33 can be challenged if the buyer takes into account such things as maintenance costs and the comparability of terminals that operate at different speeds.

Although it is harder to give positive advice, in our own case we like both thermal printing terminals (which run at 300 baud, a reasonable speed for reading, and are pleasant and relatively easy to maintain), and the graphic terminals, which have the same advantages plus the very valuable capacity of drawing pictures under computer control. I believe that the future for almost all educational use lies with graphics.

Myth 6: Computers Are Too Expensive To Use In Teaching

This issue is one of bookkeeping. With any new technology it is hard to know how to calculate costs, and computer centers in practice do this in quite different ways. (For example, existing time sharing systems quote a wide variety of prices, from .25 per hour to $50.00 per hour.) Furthermore, it is hard to make comparisons with the costs of other components of education, since these often reflect very different types of bookkeeping. It has been claimed, for example, that it costs more than $10.00 each time a book is checked out of a library. Clearly if we regarded this as a direct cost of education, libraries might be considered too expensive, yet almost no institution takes that attitude today. I think that computers are probably now competitive with other teaching methods, but this is difficult to demonstrate.

Regardless of what one thinks about the costs today, the future situation is clear: of all the cost involved in the educational process, computer costs are almost the only ones going down in price. Thus teachers, books, buildings, and films are going up in costs while computer costs, because of a rising curve of technological development, are still diminishing dramatically; so the computer will become more and more competitive as a teaching device over the next few years.

Myth 7: If We Acquire A CAI Language, That Solves Our Problems

Many computer directors take the approach that if some language is available (i.e., on the computer in an operational form) for assisting teachers in developing student computer dialogs, they have discharged their duties to the teaching community. At one time COURSEWRITER was highly promoted in this way. Today PLANIT and TUTOR tend to be the ones that are advertised as available.

Experience in many teaching applications shows, however, that the availability of a dialog language, no matter how good, is only a small part of the process of getting reliable and educationally useful teaching materials on the computer. The whole problem of an authoring system -- the way one persuades teachers to write materials, the full facilities provided, the incentives for doing this, the use of secretaries, programmers, and other kinds of auxiliary people, the testing procedures, the gathering of feedback, and the preparation of suitable computer-related text material -- is enormously more important than the question of the language itself.

Myth 8: PLATO And MITRE Are Solving All The Problems

The two large-scale projects now using the computer for learning, with massive government support, are PLATO at the University of Illinois and the Mitre Corporation project with courseware centered at Brigham Young University. Both have very interesting projects. To regard them as exhausting all the possibilities, of taking care of every eventuality, however, is quite wrong. Many interesting teaching materials that exist today could not be run on either of these systems. I think it would be unfortunate if the success or failure of these two large projects dictated all further educational use of the computer. In this regard I agree completely with Arthur Luehrmann's evaluation at the Spring Joint Computer Conference in 1972. We need a thousand flowers, not just a few! So I hope that we will not all jump onto the large projects' band-wagon.

Myth 9: Valid Educational Material Can Be Developed Without Involving Experienced Teachers In The Area

Teaching is still teaching whether done by computer or by any other device. My experience shows that really effective educational materials are still coming, in spite of talk to the contrary, almost entirely from those who are very much involved in the teaching process. The intellectual structure of every discipline is different, and the tough question of fundamental goals cannot be resolved in any simple, quick way. While computer scientists and educational psychologists can help develop learning material, I do not believe they can do it alone.

Myth 10: The Computer Used Educationally Uses Only Minor Amounts Of Computer Resources

It is often said that student use of computers requires little core, and little CPU time. While some materials exist for which this is the case, many existing examples indicate quite a contrary situation. Some of our more effective teaching programs at Irvine are very long -- some are more than 200,000 words in length and so depend heavily on overlay structures. And some are extremely demanding of the computer in terms of computational and I/O facilities. In some cases these demands exceed the abilities of current time sharing, and so some programs look toward faster systems of the future. So planning of computer use in teaching under the assumption that the minimal computer resources are required is dangerous.

One undoubtedly could proceed a bit further from these myths to other commonly held misconceptions. But I have indicated some of the more important ones to take into account for the future. In spite of these myths, the future is promising for the computer as a learning device.

UNTIL THE UPGRADE -- MAKING DO

L. D. Misek
Vassar College
Poughkeepsie, New York

Introduction

Many of us -- who now in the seventies teach computer applications within modestly equipped but scholastically stable Liberal Arts settings -- find ourselves speaking prescriptively, and with a sense of clear structure, about past projects which honest recall would have to reconstruct as having been less than "well organized." In order to arrange discussion topics in a concise, learnable, and, we hope, memorable sequence, we use hindsight to impose a certain elegance upon our recollections of the planning, testing, and judgmental phases of our own past efforts.

Where doctoral research, for example, was undertaken at a fairly large sized University endowed with a variety of experimental and constantly altering computer constellations, along with relatively hospitable policies toward "do it yourself-ers," but few formal courses in humanistic computing, trial and error was a most frequent handmaiden to rational action. Whether applying computational techniques to literary, psychological, philosophical, or other types of scholarly (but not directly mathematical) pursuits, an involvement in applications (or in applications-software) as opposed to systems programming, for instance, seemed to deal to us a more than common measure of uncertainty. One was neither mercilessly clamped into a single small system unsuited to the work, nor mercifully assured that the given systems which were up, running, and accessible a given day would be available the next -- due to exigencies of run-cues, non-commercial maintenance, and funding. In the late sixties (as, perhaps, in the fifties for hardware and the early sixties for housekeeping software) applications-users were unwise to identify their research goals too closely with particular devices.

There were few signposts to the "right" machines to handle the "right" jobs; indeed, then as today, defining goals was quite a job! Then as today, the applications-user had to improvise. In special ways according to our special tasks, we developed a hard sense of how to make do.

Even this description, which I have attempted to depict as rough and ready rather than adept and sleek, conveys a false romanticism. Making do often meant starting again. Projects were commonly scrapped in mid-course; deadlines were scandalized; counselors wielding "schedules" were frought with frustration. I do concede, then, a real distinction between making do, and really doing well. And most of the results we use in lectures or report at conclaves represent what we feel we did relatively well once we had isolated the most useful ways to use machines, and buckled down to do the work. We arrived there, however, by trying out a lot of things.

The knack of making do which represented adaptation to abundant but mutable and non-specific resources can also be a major asset in a small Computer Center with severe restrictions on resources and a clear cut need to upgrade. It can even be a staple.

In speaking of the 'small college Computer Center', I refer not to the size of the school, certainly, but to the status of the Center. In a sweeping simplification argued by the brevity of my text, I presume the small Center is one marked in some vital aspect as functioning in an ex-officio capacity. Either as majorless, interdisciplinary, non-credit, experimental, or entirely elective -- or in any combination but usually at least one of these "informal" capacities -- most small Centers operate on borrowed credits or on borrowed equipment or on borrowed time. Any such system should be expected to need an upgrading, a pulling together, or an expanding; whatever will most successfully integrate the Center into the community (scholastically) while maintaining the autonomy of its preserves and talents.

The types of needs for upgrading of hardware and software (not including enlargement of personnel structures to program, teach, research, and oversee) are almost too various to mention in summary fashion. But, on-line input stations, off-line data list and punch devices, versatile print-chains, core memory expansion, partitions for multiple users, complex-level compilers, and load reducing auxiliary storage are among them. Whether the 'very small' Center is attempting to graduate from desk-top memoried/calculators to rented terminals, or the 'medium small' Center from hard-copy teletypes to cathode-ray displays, or the 'ample' small Center from single-user batch processing to parallel batch or time-sharing augmentations (or... the reverse!) -- the under-expanded facility which is unable to meet a good portion of user demands, or to increase the number of its users in accordance with demand, is equally as needful of the pioneering zest for invention as is the affluent.

Where material resources abound, an aggressive, exploratory spirit may be expected to prevail by dint of ample opportunity. But in the minimally furnished setting, several types of apprehension militate against inventiveness, and often foster a malaise or disaffection which inhibits growth and learning.

I. Glorified Pasts and Utopian Futures

What are some of the factors which can inhibit a creative and ingenious use of resources while waiting for upgrades?

(a) I have mentioned the likelihood that classroom teachers may quite innocently limit the perspectives of their students by romanticizing prior efforts launched at larger installations. It seems natural that those who teach computer applications in an academic setting will show preferences for certain types of equipment based on personal experience in their own research. In the small Center, it is likely that the research backgrounds of the personnel -- acquired in the context of the University advanced degree program, or within the Computing Industry itself -- will vary greatly and reflect diverse concerns. Based on the understandable biases which accompany familiarity, and, hopefully, some degree of former (if sometimes exaggerated) "success" on a system, we can expect to find some time-sharing and some card-system buffs at the lecterns; corresponding systems, however, might not be in use at the Center.

(b) One substantial danger in this type of nostalgia is the direct effect that students captivated by 'ideal' portrayals of other systems, but somewhat naive of their pitfalls, may despair of doing interesting work on the equipment in their Center. Or, indirectly, and not consciously on their part or on that of their teachers, they may be encouraged to develop a similar reverence for particular devices currently available, to the exclusion of the other systems or materials, however modest, which are also present in that very Center. If a teacher, then, seems to have "survived" solely because of a particular system, the student may develop parallel rigidities in attitude, unwarranted dependencies on certain systems without ample exploration.

Sadly, whatever the system status or size, internal snobberies and loyalties to only certain devices or programming languages can bring on premature stylizations and stifle creativity. On card systems, for example, users can control the rate of input preparation in accordance with their schedules (for producing cards); line-printer output, even if voluminous, is rapid; and intermediate processing is rarely open to interruption. On time-sharing terminals, speedy and contiguous input is desirable; output is "relatively" slow on the character-printer; but, intermediary processing can, interactively, be interrupted or rerouted by the user at his keyboard. The batch, or card, system is optimal where the output document is lengthy and "independent" of context; time-sharing serves the user best where the output production is not great in length, and highly sensitive to (interactive) context. Yet, there are time-sharing afficionados who will sit before a character printing terminal (TTY or CRT) for twenty-four hours printing a "key word in context" concordance, and others who will attempt to program extensive Computer Aided Instruction sequences on card systems, waiting hours, days, or even a week for the printing of the next brief teaching "frame," even though both batch and time-sharing systems are available in the same setting. Now, in the spirit of creativity, such stout hearted attempts might be applauded up to the point where health gives out, and the Computer Center gains an ugly reputation for wearying its students out of doing other coursework. The student who has not frozen his interests onto a particular device, but, rather, stayed 'light on his feet', will thankfully give up his pet machine, abandon whimsy, and move on, reshaping goals or switching to new systems.

(c) If we can characterize the two types of bias reflected in classroom instruction and student response as prejudicial against systems "Not Invented Here" (or there!) (NIH/T) -- then we might speak of fixation on the unattained as representing the lament of "Not Invented Yet!" (NIY). This is the treacherous conviction that only future systems can resolve present problems. Whether "future" stands here for "larger" or just somehow "better" systems, we are all familiar with the plaint that researchers cannot begin to explore fields of inquiry...students cannot fulfill their assignments...teachers cannot even preview a topic ...because only the proverbial machine 'around the corner' will suffice. The humble dodging of productive use of an environment as it now stands can very well be innocent, and is most often tinged with fact, since industry announces systems far more grand than we are likely to enjoy in academe. In all fairness: programming instructors do often find their lectures confounded by the present levels of compilers; text-processing teachers are frequently circumscribed by diminutive on-line storage, and minimal disk, drum, or off-line reserves. But are these limits not the hard realities around which the user must maneuver to be able to explore, if not complete, sophisticated or broad projects?

(d) Unfortunately, it is particularly in regard to faith in future prospects that the literature can be misinterpreted to suggest that present efforts not be modified, but virtually halted pending the advent of millennia. There is a euphoria found in the scholarship on user-applications, which can wash out the advantages of what is now at hand. Who would deny, as we read humanistic journals, that a fully expanded character-set (upper/lower-case with underlining, boldface, italics, and even multiple type-fonts) could be superior to the upper-case alphabet alone, with just a sprinkling of punctuation symbols? Who might not prefer to teletypes, on-line upper/lower-case display screens with graphics, hard-copy options, and a spooling of extensive output to run background or third-shift on line-printers? Scholars who describe such systems often give the users in small Centers a dire sense of poverty, labouring as they are over 026 or 029 keypunch machines, and this at times in a separate building (library or technical department) physically removed from the Computing Center. Where some scholars extol the virtues of particular advanced devices, even more utopian are descriptions of the mini-machine rooms, with tape-casette pre-processing data depots, CRT graphics output alternating with line-printers (upper/lower-case, of course) and on-line simultaneous production of magnetic tapes which drive a photo-typesetting device.

II. Practical Presents

I will proceed on the assumption that while titillating, optimistic, and mouth-watering, highly expanded user systems are not likely to be very soon bestowed on the small Center, which I have defined as genuinely needful of an upgrade at the level of its "staples." Again, I speak not of an engineering subgroup in an amply endowed, smallish school, but of a small and growing Center in a non-technical setting -- one in which a "dedicated" system under funded research is not presently developing.

I refer to the bootstrapping Center, modest in equipment, but imbued with faith and fervour...arising from experiment, an academic venture first initiated through the Payroll people (in nerve, and temperament, a class of saints!)...springing from Mathematics...flirting with Humanities..converting by inches...tolerating...being tolerated...influxing...emerging with a humanistic ethos paralleling the statistical...finally integrating as full-fledged, somewhat "established," credit-giving course complexes in their native colleges...attracting students, teachers, researchers, whose numbers and enthusiasms strain the systems, schedules, personnel or budget...and eventually experiencing a need to upgrade and expand.

What are some practical ways, without recourse to bias or escapism, for the 'small' Center user to make do while waiting for upgrades?

He can SEARCH for a suitable medium, by experimenting with a small sample of relevant data, encoding and processing it on each of the available systems within the Center.

He can SUBSTITUTE where necessary, by creating code-conventions to stand in for missing symbols, and abbreviations or notations to place-hold for lengthy terms and complex concepts. He can break up lengthy programs into separate modules brought successively to memory from disk or tape to process serially altered data groups.

He can MODEL, by scaling down projects initially, to provide the opportunity to finish thorough pilot studies. If text-processing must be accomplished on a TTY, he can narrow the line length or KWIC fields; if batch must be used for CAI performance, he can find ways to use single "frames" to illustrate if not perfect the feedback precept more easily implemented on an interactive system.

He can IMPROVISE, by splitting up projects among different systems, using hybrid methods to process a project by "chunks," alternatively on batch or time-sharing (and by hand or using off-line assistance through printers and sorters).

Finally, he can ABANDON the project which seems doomed for want of specific devices, just at that happy stage where a progress report and prospectus can be as illuminating as a relentless, ill fated, dogged attempt to 'see through' the unfeasible.

These are only very local examples, of course, of the types of making do which are invented daily in the small Computer Center. More inviting of discussion are some general, or theoretical, considerations (cautions and assurances) concerning coming systems and our common need to upgrade. Offered merely as the observations of just one investigator, they invite readers' expansions:

1. Upgrading can be crucial.

By the very self-help nature of the small Computer Center, most improvements will mark giant steps. No system which functions for only a small fraction of applicant users -- no matter how talented or productive this group may seem -- can fairly serve the academic setting. If, for example, a dozen students need vie for one keypunch, wait all night to snatch their time at terminals, or consistently (they claim) miss meals, "social life," and preparation for their courses (a touch of poverty is, of course, starch for the soul!) -- then the system surely must be substandard. It would be folly to expect the teacher to make interesting and instructive a field of study in which supporting equipment or software failed consistently, and, accordingly, a sense of superficiality or sheer futility spread wide among the students. It would be unwise to expect that an instructor could embody perseverance, wit, and industry, if scheduling or budgeting prohibited more advanced investigations than were normally presented in the classroom lecture. It would be ultimately unfeasible to simulate, through overly lengthy or complex chains of programs and procedures, the essential strategies and the economies which make computing meaningful.

2. Until you do well, make do.

For the students who do have present access -- in pursuit of programming or applications -- we as teachers in small Centers can provide examples of creative usage and inventiveness. In a literary study, as an instance, where analysis of textual style is paramount, data cards sorted by hand on a table can be preferable to on-line sorts discarded due to access problems. For the programmer, the sort cannot be bypassed, since the sorting algorithm, programmed and debugged on-line, comprises the essential task; but, here, the actual use of program output for language study can be summarized rather than extensively developed. Certain stages in a

project, generally, can be suggested, sketched, or simulated, with one crucial phase allotted closer treatment in accordance with resources.[1]

3. Relevance can be modeled.

If we, as teachers, wish to communicate -- and wish our students to appreciate -- the significance of computationally assisted studies, we are more likely to succeed through a few well considered results (admittedly qualified by limited tools of inquiry) than we are by ceaseless explanations as to how a technology can both be a marvelous aid for the scholar, and, at the same time, somehow not quite ready for use.

4. The future is wily.

It seems most unlikely that future systems will be addressed to problems as we presently conceive them. Novelty invites distraction, and with an upgrade we will no doubt find new problems to worry about and new projects to dream on; find that our deadlines have passed; and, hopefully, find that our goals have matured. Therefore, we are obligated to meet present problems in the present.

Conclusion

I rest my case for making do on the following quotes, which, though written from the point of view of business applications, seem to me especially prophetic for the academic setting. I hope that these remarks might wink out at my audience as they do at me, expressing some of the lighter cruelties of the profession, our dissatisfaction with the 'present' and with what we might call 'future presents':

"Soon after the computer became technically reliable, it also became obsolete. For better computers were now on the market."[2]

[but...]

"Each succeeding generation of computers seems to teach businesses to operate slower faster."[3]

Might we not apply the ironies of business applications to horizons in the Liberal Arts? My main if brief thesis here is that we often await software and equipment which may allow us in the future to complete in part the tasks we fully comprehended yesterday. This I take to be the flavor of the first quote.

At the same time (second quote), along with the expansion of our resources we can expect to have delivered to us elongations of our own goals: typical revisions and revampings which (with all the circularity of Sisyphus, and the tormenting stone which is most likely to descend the more it climbs the peak) force upon us the dilemma -- can we ever reach the "top"?

My recommendation has been that we make do in our own time. Most small Computer Centers do enjoy the privilege of having, at least in some areas, facilities expanded beyond those of other Centers with equivalent educational concerns. It seems imperative to be as productive as we can in un-upgraded work environments, to think as creatively as possible within the present, to avoid chauvinisms and 'device fixations', to encourage adaptation, and accept needed improvements to our systems (should the funds begin to flow) as fortunate returns on prayers, and not essentials in our plans.

Notes

[The author is Assistant Professor, Computer Science Studies (headed by Professor Winifred A. Asprey) at Vassar College, Poughkeepsie, New York. Vassar's primary computing resources are a 32K 360/30 single-user batch (card) system, four remote APL-terminals into IBM equipped facilities in Poughkeepsie, and a genius of a Director. Ideas discussed here are a composite of impressions drawn from present teaching experience at Vassar, and earlier investigations at other institutions. Acknowledgment is especially due Prof. Edward L. Glaser, Chairman, Case Western Reserve University Department of Computing and Information Science; Stanley Y. Curry, past President, Chi Corporation; Dr. Robin B. Lake of the CWRU School of Medicine Department of Biometry; and Michael Luton, Director, CWRU Center for Continuing Education; for enriching opportunities as Graduate Assistant, Consultant, Research Associate, and Instructor, respectively. References: "Automated Contextual Analysis of Thematic Structure in Natural Language" (A.R. Jennings Report 1103, CWRU, 1970); "Chronology and Character: a Computer-Assisted Study of Satan in Paradise Lost" (1971 Midwestern Modern Language Association Conference); Computing a Context: Style, Structure, and the Self-Image of Satan in Paradise Lost (1971 doctoral dissertation, CWRU, under Dr. John S. Diekhoff); Context Concordance to Paradise Lost (A.R. Jennings Computing Center, CWRU, 1971)].

[1]In one Vassar marathon, five students jointly keypunched a 2000-word card dictionary, and hand sorted this box-length deck in two hours, having estimated that the intricacies of human communication and the required transportation of data could take two weeks on the system. Since this stage of psycholinguistic project in automated recognition (of semantic patterns) seemed largely clerical, the amount of potential learning involved in automation at that stage seemed to them inequitable. They were in this particular case pursuing results as defined by their goals (and achieving them!), through the hand/eye coordination which became encoded as, generically, "gorilla work." Later in life they will encounter the method professionally dubbed as "quick and dirty."

[2]Mumford, Enid, and Banks, Olive. The Computer and the Clerk (London: Routledge and Kegan Paul, 1967), p. 59, a description of initial stages in the implementation of computing facilities at the Royal Exchange Bank.

[3]Smith, Paul T. Computers, Systems, and Profits (New York: American Management Association, 1969), Dedication.

Appendices

The following three pages illustrate conversion of a context-concording algorithm from a large-scale computing facility to a small College Center. The Context Concordance to Satan in John Milton's Paradise Lost is represented in three formats:

A. (1969-71).

Upper/lower-case Harris Intertype FOTO transformation from a magnetic tape produced on CWRU/Chi Corporation's Univac 1108 (132K) running under Exec-IV. The original program, CRIC-8 ("Cross-Reference-in-Context-for the 1108") was coded in Algol-60 by William Cornwell (Case, '71) with L. D. Misek. CRIC-8 runs as one module, in approximately 2.1 minutes, for the roughly 8000-word data-base.

B. (1972-73).

Adaptation of CRIC-8 for a smaller system, in a sequence of separately executed program modules, coded in PL/1 by Thomas Mylott III (Vassar, '74) with L. D. Misek, as the "Vassar Concordance Generator" (VCG). While only samples of the data are represented here, the complete 8000-word data-base (apx.) was context-concorded and printed in just under 2.5 hours, on an IBM 360/30 (32K) system, dumping the output from mag-tape to printer. The all upper-case (48 character) print-chain is augmented by coding conventions sufficient to transcribe all punctuation and special symbols found in the original.

C. (1973).

Time-sharing adaptation of context-concording algorithms, seeks and displays words selectively. Program in APL is written by Rexford Swain (Computer Center Intern, Vassar, '74) with L. D. Misek. Vassar's four APL-terminals tie in with S.E.C.O.S., a Poughkeepsie based not-for-profit educational support group (Rebecca Willis, President).

APPENDIX A.

etlev'el so main to our success.	I	bring, • Which of us who beho	7	HVN—6	SAT/RL	TO NSRCH+	1	1	4	471	7
rn ABt back Perverse. • But that	I	doubt, nowever witness Heave	6	HVN—6	SAT/RL	TO FOLWRS	2	2	5	583	3
• For joy of offer'd peace: but	I	suppose • If our proposals on	9	HVN—6	SAT/RL	TO FOLWRS	3	3	6	617	7
n his rage • Can else inflict do	I	repent or change, • Though ch	10	HELL—1	SATAN	TO BEELZ	2	1	1	96	5
s perhaps • Shall grieve him, if	I	fail not, and disturb • His t	11	HELL—1	SATAN	TO BEELZ	3	2	2	167	6
even[2]. • What matter where, if	I	be still the same, • And what	12	HELL—1	SATAN	TO BEELZ	4	3	3	256	5
I be still the same, • And what	I	should be, all but less than	12	HELL—1	SATAN	TO BEELZ	4	3	3	257	3
our, though oppress and fall'n, •	I	give not Heaven[2] for lost.	15	HELL—2	SATAN	TO COUNCL	1	1	1	14	1
engere and se herd escape. • But	I	should ill become this Thron	15	HELL—2	SATAN	TO COUNCL	2	2	2	445	2
• from attempting Wherefore do	I	assume • These Royalties, an	16	HELL—2	SATAN	TO COUNCL	2	2	2	450	8
• Against a wakeful [illegible], while	I	abroad • Through all the coas	16	HELL—2	SATAN	TO COUNCL	2	2	2	463	8
• To yonder Gates? th gh them	I	mean to pass, • That be assur	17	GATE—2	SATAN	TO DEATH	1	1	3	684	4
de • What it intends, till first	I	know of thee, • What thing th	18	GATE—2	SATAN	TO SIN+OT	1	1	4	740	6
that Phantasm call'st my Son? •	I	know thee not, nor ever saw t	18	GATE—2	SATAN	TO SIN+OT	1	1	4	744	4
unforeseen, unthought-of, know •	I	come no enemy, but to set fre	19	GATE—2	SATAN	TO SIN+OT	2	2	5	822	1
with us from on high from them	I	go • This uncouth errand sole	19	GATE—2	SATAN	TO SIN+OT	2	2	5	826	9
n this more secret now design'd,	I	haste • To know, and this onc	19	GATE—2	SATAN	TO SIN+OT	2	2	5	858	7
yes, • Chaos and ancient Night,	I	come no spy, • With purpose t	20	CHAO—2	SATAN	TO NGT+CH	1	1	6	870	5
e, and without guide, half lost,	I	seek • What readiest path lea	20	CHAO—2	SATAN	TO NGT+CH	1	1	6	975	7
esses lately, thither to arrive •	I	travel this profound, direct	20	CHAO—2	SATAN	TO NGT+CH	1	1	6	980	1
e it brings • To your behoof, if	I	that Region lost, • All usur	20	CHAO—2	SATAN	TO NGT+CH	1	1	6	982	5
orbs his choice to dwell; • That	I	may find him, and with secret	21	SUN—3	SATAN	TO URIEL	1	1	1	671	2
• their diminisht heads, to thee	I	call, • But with no friendly	22	ERTH—4	SATAN	TO SELF.1	1	1	1	35	7
name • O Sun, to tell thee how	I	hate thy beams • That bring t	22	ERTH—4	SATAN	TO SELF.1	1	1	1	37	7
my remembrance from what state •	I	fell, how glorious once above	22	ERTH—4	SATAN	TO SELF.1	1	1	1	39	1
• From me, whom he created what	I	was • In that bright eminence	22	ERTH—4	SATAN	TO SELF.1	1	1	1	43	7
t but malice, lifted up so high •	I	'sdein'd subjection, and thou	22	ERTH—4	SATAN	TO SELF.1	1	1	1	60	1
o owe, • Forgetful what from him	I	still receiv'd, • And underst	22	ERTH—4	SATAN	TO SELF.1	1	1	1	64	5
ein'd • Me some inferior Angel,	I	had stood • Then happy; no un	22	ERTH—4	SATAN	TO SELF.1	1	1	1	69	6
• Me miserable! which way shall	I	fly • Infinite wrath, and inf	22	ERTH—4	SATAN	TO SELF.1	1	1	1	73	8
nd infinite despair? • Which way	I	fly to Hell; myself am Hell	22	ERTH—4	SATAN	TO SELF.1	1	1	1	76	8
opens wide, • To which the Hell	I	suffer seems • Heaven[2] s	22	ERTH—4	SATAN	TO SELF.1	1	1	1	78	6
among the Spirits beneath, whom	I	seduc'd • With other Promises	22	ERTH—4	SATAN	TO SELF.1	1	1	1	83	6
aunts • Then to submit, boasting	I	could subdue • The[2] Omnipo	22	ERTH—4	SATAN	TO SELF.1	1	1	1	85	5
e, they little know • How dearly	I	abide that boast so vain, • U	22	ERTH—4	SATAN	TO SELF.1	1	1	1	87	3
, • Under what torments inwardly	I	groan: • While they adore me	22	ERTH—4	SATAN	TO SELF.1	1	1	1	88	6
high advanc't • The lower still	I	fell, only supreme • In misar	22	ERTH—4	SATAN	TO SELF.1	1	1	1	91	4
h joy Ambition finds. • But say	I	could repent and could obtain	22	ERTH—4	SATAN	TO SELF.1	1	1	1	95	3
e. • And heavier fall: so should	I	purchase dear • Short interm	22	ERTH—4	SATAN	TO SELF.1	1	1	1	101	8
as far • From granting he[1], as	I	from begging peace • All hop	22	ERTH—4	SATAN	TO SELF.1	1	1	1	104	6
Empire with Heaven's[3] King	I	hold • By thee, and more than	22	ERTH—4	SATAN	TO SELF.1	1	1	1	111	8
el no purpose'd foe • To you whom	I	could pity thus forlorn • Tho	23	ERTH—4	SATAN	TO SELF.2	1	1	2	374	4
could pity thus forlorn • Though	I	unpitied. League with you I	23	ERTH—4	SATAN	TO SELF.2	1	1	2	375	2
gh I unpitied: League with you	I	seek, • And mutuel amity so e	23	ERTH—4	SATAN	TO SELF.2	1	1	2	376	7
mity so strait, so close, • That	I	with you must dwell, or you m	23	ERTH—4	SATAN	TO SELF.2	1	1	2	377	2
r'e work; he gave it me, • Which	I	as freely give: Hell shall u	23	ERTH—4	SATAN	TO SELF.2	1	1	2	381	2
or him who wrong'd. • And should	I	at your harmless innocence •	23	ERTH—4	SATAN	TO SELF.2	1	1	2	388	3
ur harmless innocence • Melt, as	I	do, yet public reason just, •	23	ERTH—4	SATAN	TO SELF.2	1	1	2	389	3
• To do what else though damn'd	I	should abhor.''	23	ERTH—4	SATAN	TO SELF.2	1	1	2	392	7
fill • Of bliss on bliss, while	I	to Hell am thrust, • Where n	24	ERTH—4	SATAN	TO SELF.3	1	1	3	508	8
as; • Yet let me not forget what	I	have gain'd • From their own	24	ERTH—4	SATAN	TO SELF.3	1	1	3	512	7
on to build • Their ruin! Hence	I	will excite their minds • Wit	24	ERTH—4	SATAN	TO SELF.3	1	1	3	522	4
? • But first with narrow search	I	must walk round • This Garde	24	ERTH—4	SATAN	TO SELF.3	1	1	3	526	8
chance but chance may lead where	I	may meet • Some wand'ring Sp	24	ERTH—4	SATAN	TO SELF.3	1	1	3	530	8

JK.24.ETL.S3.0.80.AS	141	ELIER CAN ENSUE*. & BUT* FIRST WITH NARROW SEARCH I* MU,T WALK ROUND & THIS* GARDEN*, AND NO CORNER E
JK.24.ETL.S3.0.80.AS	143	E UNSPI'D,. & A* CHANCE BUT CHANCE MAY LEAD WHERE I* MAY MEET & SOME* HAND'RING SPIRIT* OF HEAV'N*, R
JK.24.ETL.S3.0.80.AS	147	VE* WHILE YE MAY, & YET* HAPPY PAIR,. ENJOY, TILL I* RETURN, & SHORT* PLEASURES, FOR LONG WOES ARE TO
JC.29.FBE.72.2.80.AS	187	& (EIE* I* MUST CONTEND,... & REST* WITH THE BEST, THE SEND
JC.30.LBE.G1.1.80.AS	192	ST IN HEAV'N* TH'I ESTEEM*I OF WISE, & AND* SUCH I* HELD THEE,. BUT THIS QUESTION ASKT & PUT* ME IN
JC.30.LBE.G1.1.80.AS	199	OMPENSE & DULE* WITH DELIGHT, WHICH IN THIS PLACE I* SOUGHT,. & TO* THEE DO REASON,. WHO KNOWST ONLY
JC.30.LBE.G1.1.80.AS	207	HAT IMPLIES NOT VIOLENCE OR HARM.)) & (ENO* THAT I* LESS ENDURE, OR SHRINK FROM PAIN, & INSULTING* A
JC.30.EBE.G1.1.80.AS	208	FROM PAIN, & INSULTING* ANGEL,* WELL THOU KNOWEST I* STOOD & THY* FIERCEST, WHEN IN BATTLE* TO THY AI
JC.30.FBE.G1.1.80.AS	217	& THROUGH* WAYS OF DANGER BY HIMSELF UNTRI'D. & I* THEREFORE, I* ALONE FIRST UNDERTOOK & TO* WING T
JC.30.LBE.G1.1.80.AS	217	AYS OF DANGER BY HIMSELF UNTRI'D. & I* THEREFORE, I* ALONE FIRST UNDERTOOK & TO* WING THE DESOLATE AR
JS.17.EBE.G3.1.80.AS	228	& (EITHER* WHEN I* AM THY CAPTIVE TALK OF CHAINS, & PROUD* LIMITARY
EE.33.EGI.S4.0.80.AS	250	AIL SUMM'D UP IN MAN*. & WITH* WHAT DELIGHT COULD I* HAVE WALKT THEE ROUND & IF* I* COULD JOY IN AUGH
EE.33.EGI.S4.0.80.AS	251	WHAT DELIGHT COULD I* HAVE WALKT THEE ROUND & IF* I* COULD JOY IN AUGHT, SWEET INTERCHANGE & OF* HILL
EE.33.EGI.S4.0.80.AS	254	OREST* CROWN'D, & RUCKS*, DENS*, AND CAVES*,. BUT I* IN NONE OF THESE & FIND* PLACE OR REFUGE,. AND T
EE.33.EGI.S4.0.80.AS	255	E OF THESE & FIND* PLACE OR REFUGE,. AND THE MORE I* SEE & PLEASURES* ABOUT ME, SO MUCH MORE I* FEEL
EE.33.EGI.S4.0.80.AS	256	E MORE I* SEE & PLEASURES* ABOUT ME, SO MUCH MORE I* FEEL & TORMENT* WITHIN ME,--AS FROM THE HATEFUL S
EE.33.EGI.S4.0.80.AS	260	ORSE WOULD BE MY STATE. & BUT* NEITHER HERE SEEK I*, NO NOR IN HEAV'N* & TO* DWELL, UNLESS BY MEAN'S
EE.33.EGI.S4.0.80.AS	263	NDR* HOPE TO BE MYSELF LESS MISERABLE & BY* WHAT I* SEEK, BUT OTHERS TO MAKE SUCH & AS* I*, THOUGH T
EE.33.EGI.S4.0.80.AS	264	& BY* WHAT I* SEEK, BUT OTHERS TO MAKE SUCH & AS* I*, THOUGH THEREBY WORSE TO ME REDOUND.. & FOR* UNL
EE.33.EGI.S4.0.80.AS	269	Y WORSE TO ME REDOUND.. & FOR* ONLY IN DESTROYING I* FIND EASE & TO* MY RELENTLESS THOUGHTS,. AND HIM
EE.33.EGI.S4.0.80.AS	276	NTRIVING, THOUGH PERHAPS & NOT* LONGER THAN SINCE I* IN ONE NIGHT* FREED & FROM* SERVITUDE INGLORIOUS
EE.33.EGI.S4.0.80.AS	293	THEIR* EARTHY CHARGE*,. OF* THESE THE VIGILANCE S I* DI'AD, AND TO FLOOR, THUS WRAPT IN MIST & OF* MI
SE.34.EGI.S4.0.80.AS	297	SE MAZY FOLDS & TO* HIDE ME, AND THE DARK INTENT I* BRING. & OF FOUL DESCENT- THAT I* WHO LATE CONTE
SE.34.EGI.S4.0.80.AS	298	THE DARK INTENT I* BRING. & OF FOUL DESCENT- THAT I* WHO LATE CONTENDED & WITH* GODS* TO SIT THE HIGH
EE.34.EGI.S4.0.80.AS	308	TER* EMELONG BACK ON ITSELF RECOILS,. & LET* IT,. I* RICK NOT, SO IT *LIGHT WELL AIM'D, & SINCE* HIGH
EE.34.EGI.S4.0.80.AS	309	ECK NOT, SO IT *LIGHT WELL AIM'D, & SINCE* HIGHER I* FALL SHORT, ON HIM WHO NEXT & PROVOKES* MY ENVY,
EE.34.EGE.S5.0.80.AS	323	*,OPPORTUNE TO ALL ATTEMPTS, & HER* HUSBANDS, FOR I* VIEW FAR ROUND, NOT NIGH, & WHOSE* HIGHER INTELL
EE.34.EGE.S5.0.80.AS	324	OUND, NOT NIGH, & WHOSE* HIGHER INTELLECTUAL MORE I* SHUN, & AND* STRENGTH, OR COURAGE HAUGHTY, AND O
EE.34.EGE.S5.0.80.AS	328	& FOE* NOT INFORMIDABLE, EXEMPT FROM WOUND, & I* NOT,. SO MUCH HATH HELL* DEBAS'D, AND PAIN & ENF
EE.34.EGE.S5.0.80.AS	329	HELL* DEBAS'D, AND PAIN & ENFEEBL'D* ME, TO WHAT I* WAS IN HEAV'N*. & SHEE* FAIR,DIVINELY FAIR, FIT
EE.34.EGE.S5.0.80.AS	334	E* WELL-FEIGN'D, & THE* WAY WHICH TO HER RUIN NOW I* TEND))
SM.35.EGE.E4.1.80.AS	338	'N* OF MILDNESS, WITH DISDAIN, & DISPLEAS'D* THAT I* APPROACH THEE THUS, AND GAZE & INSATIATE*, I* TH
SM.35.EGE.E4.1.80.AS	339	HAT I* APPROACH THEE THUS, AND GAZE & INSATIATE*, I* THUS SINGLE, NOR HAVE FEAR'D & THY* ANGEL SOLE,
SM.36.EGE.E5.1.80.AS	356	COMMAND'ST, AND MIGHT THOU SHOULDST BE OBEY'D.. & I* WAS AT FIRST AS OTHER BEASTS* THAT GRAZE & THEE*
SM.36.EGE.E5.1.80.AS	359	NOTHING HIGH.. & TILL* ON A DAY ROVING THE FIELD, I* CHANC'D & A* GODDLY TREE* FAR DISTANT TO BEHOLD
SM.36.EGE.E5.1.80.AS	362	UIT OF FAIREST COLOURS MIXT, & RUDDY* AND GOLDE*.. I* NEARER DREW TO GAZE,. & WHEN FROM THE BOUGH I S
SM.36.EGE.E5.1.80.AS	368	T TEND THEIR PLAY, & TO* SATISFY THE SHARP DESIRE I* HAD & OF* TASTING THOSE FAIR APPLES*, I* RESOLV'D
SM.36.EGE.E5.1.80.AS	369	P DESIRE I* HAD & OF* TASTING THOSE FAIR APPLES*, I* RESOLV'D & NOT* TO DEFER,. HUNGER AND THIRST AT
SM.36.EGE.E5.1.80.AS	373	UIT, URG'D ME SO KEEN. & ABOUT* THE MOSSY* TRUNK* I* WOUND ME SOON; & FOR HIGH FROM GROUND THE BRANCH
SM.36.EGE.E5.1.80.AS	380	& TEMPTING SO NIGH,--TO PLUCK AND EAT MY FILL & I* SPAR'D NOT, FOR SUCH PLEASURE TILL THAT HOUR & A
SM.36.EGE.E5.1.80.AS	381	TILL THAT HOUR & AT* FEED* OR FOUNTAIN* NEVER HAD I* FOUND. & SATED* AT LENGTH, ERELONG I* MIGHT PERC
SM.36.EGE.E5.1.80.AS	382	* NEVER HAD I* FOUND. & SATED* AT LENGTH, ERELONG I* MIGHT PERCEIVE & STRANGE* ALTERATION IN ME, T
SM.36.EGE.E5.1.80.AS	387	& THENCEFORTH* TO SPECULATIONS* HIGH OR DEEP & I* TURN'D MY THOUGHTS, AND WITH CAPACIOUS MIND & CO
SM.36.EGE.E5.1.80.AS	392	CE*, AND IN THY BEAUTY'S* HEAV'NLY RAY* & UNITED* I* BEHOLD,. NO FAIR TO THINE & EQUIVALENT OR SECON
SM.37.EGE.E6.1.80.AS	401	MYRRH* AND BALM*,. IF THOU ACCEPT & MY* CONDUCT, I* CAN BRING THEE THITHER SOON.))
SM.39.EGE.E8.1.80.AS	406	(SOON-GIVING* PLANT*, & MOTHER* OF SCIENCE*, NOW* I* FEEL THY POWER* & WITHIN* ME CLEAR, NOT ONLY TO
SM.39.EGE.E8.1.80.AS	436	S THEY KNOW. & THAT* YE SHOULD BE AS GODS*, SINCE I* AS MAN, & INTERNAL* MAN*, IS BUT PROPORTION MEET
SM.39.EGE.E8.1.80.AS	438	& INTERNAL* MAN*, IS BUT PROPORTION MEET, & I* OF BRUTE HUMAN, YEE OF HUMAN GODS*. & SO YE SHA
SM.39.EGE.E8.1.80.AS	446	& ON* OUR BELIEF, THAT ALL FROM THEM PROCEEDS,. & I* QUESTION IT, FOR THIS FAIR EARTH* I* SEE, & WARM'D
SM.39.EGE.E8.1.80.AS	446	PROCEEDS,. & I* QUESTION IT, FOR THIS FAIR EARTH* I* SEE, & WARM'D BY THE SUN*, PRODUCING EVERY KIND
LL.40.LBE.D4.2.80.AS	461	NOW HAVE GIV'N TO BE THE RACE* & OF* SATAN* (FOR I* GLORY IN THE NAME & ANTAGONIST* OF HEAV'N'S* ALM
LL.40.LBE.D4.2.80.AS	469	TINENT* & OF* EASY THOROUGHFARE. THEREFORE* WHILE I* & DESCEND* THROUGH DARKNESS,* ON YOUR ROAD* WITH
LL.40.LBE.D4.2.80.AS	472	YOUR THRALL, AND LASTLY KILL. & MY* SUBSTITUTE* I* SEND YE, AND CREATE* & PLENIPOTENT* ON EARTH,* O
NA.41.LEP.E5.N.AD.AS	487	& FOR* IN POSSESSION, SUCH, NOT ONLY A PLOT, & I* CALL YE AND DECLARE YE NOW, RETURN'D & SUCCESSFU
NA.41.LEP.E5.N.AD.AS	495	PERIL GREAT ACHIEV'D. LONG* WERE TO TELL & WHAT* I* HAVE DONE, WHAT SUFFER'D, WITH WHAT PAIN & VOYAG
NA.41.LEP.E5.N.AD.AS	499	IS PAY'D & TO* EXPEDITE YOUR GLORIOUS MARCH,. BUT I* & I TIED* PUT MY UNCOUTH PASSAGE, FORCE* TO RIDE
NA.41.LEP.E5.N.AD.AS	505	S UPROAR & PROTESTING* FATE* SUPREME,. THENCE HOW I* FOUND & THE* NEW CREATED WORLD*, WHICH FAME IN H

```
'HY NOT? SOME OTHER POWER AS GREAT MIGHT HAVE
   HEAV'N'S MATCHLESS KING: AH WHEREFORE! HE
 E DESERV'D NO SUCH RETURN FROM ME, WHOM HE
 IGHT ALL THE STARS HIDE THEIR DIMINISHT HEADS
 WORSE AMBITION THREW ME DOWN WARRING IN HEAV'N
   THREW ME DOWN WARRING IN HEAV'N AGAINST HEAV'N'S
WHOM WAS I THOU THEN OR WHAT TO ACCUSE, BUT HEAV'N'S
   NOW KNOWS; AT WHOSE SIGHT ALL THE STARS HIDE
 ME, AND WROUGHT BUT MALICE; LIFTED UP SO HIGH
'SDAIN'D SUBJECTION, AND THOUGHT ONE STEP HIGHER
 AND THOUGHT ONE STEP HIGHER WOULD SET ME HIGHEST
   HARD. WHAT COULD BE LESS THAN TO AFFORD HIM
   PRAISE, THE EASIEST RECOMPENSE, AND PAY HIM
PRAISE, STILL TO OWE; FORGETFUL WHAT FROM HIM
   I WAS IN THAT BRIGHT EMINENCE, AND WITH HIS
AND WITH HIS GOOD UPBRAIDED NONE; NOR WAS HIS
   AND PAY HIM THANKS, HOW DUE! YET ALL HIS
 AND DISCHARG'D; WHAT BURDEN THEN? O HAD HIS
HAVE ANGEL'D, AND HE THOUGH MEAN DRAWN TO HIS
   FREE LOVE DEALT EQUALLY TO ALL? BE THEN HIS
   I HAD STOOD THEN HAPPY; NO UNBOUNDED HOPE
   AND ADD THY NAME O SUN, TO TELL THEE HOW
TO MY REMEMBRANCE FROM WHAT STATE I FELL, HOW
   EASIEST RECOMPENSE, AND PAY HIM THANKS, HOW
                    ----------
STARS HIDE THEIR DIMINISHT HEADS; TO THEE I
  AND ADD THY NAME O SUN, TO TELL THEE HOW I
 BRING TO MY REMEMBRANCE FROM WHAT STATE I
SUCH RETURN FROM ME, WHOM HE CREATED WHAT I
AND WROUGHT BUT MALICE; LIFTED UP SO HIGH I
   STILL TO OWE; FORGETFUL WHAT FROM HIM I
DESTINY ORDAIN'D ME SOME INFERIOUR ANGEL, I
   HEAVEN, HOW DUE! YET ALL HIS GOOD PROV'D ILL
AND DISCHARG'D, AND IN A MOMENT QUIT THE DEBT IMMENSE
   AND WORSE AMBITION THREW ME DOWN WARRING IN
REMOVE FROM ME, WHOM HE CREATED WHAT I WAS IN
   NOW DUE! YET ALL HIS GOOD PROV'D ILL IN
 ONE STEP HIGHER WOULD SET ME HIGHEST, AND IN
BY OWING OWES NOT, BUT STILL PAYS, AT ONCE INDEBTED
 WAS HIS POWERFUL DESTINY ORDAIN'D ME SOME INFERIOUR
ACCURST, SINCE LOVE OR HATE, TO ME ALIKE, IT
                    ----------
   IN HEAV'N AGAINST HEAV'N'S MATCHLESS KING
                    ----------
   NOR WAS HIS SERVICE HARD. WHAT COULD BE LESS
PROV'D ILL IN ME, AND WROUGHT BUT MALICE; LIFTED
   CROWN'D, LOOK'ST FROM THY SOLE DOMINION LIKE
O THOU THAT WITH SURPASSING GLORY CROWN'D, LOOK'ST
THEN IN WHAT TO ACCUSE, BUT HEAV'N'S FREE LOVE
   LOVE DEALT EQUALLY TO ALL? BE THEN HIS LOVE
   TO ALL? BE THEN HIS LOVE ACCURST, SINCE LOVE
                    ----------
HIS GOOD PROV'D ILL IN ME, AND WROUGHT BUT MALICE
ME DOWN WARRING IN HEAV'N AGAINST HEAV'N'S MATCHLESS
```

```
ASPIR'D, AND ME THOUGH MEAN DRAWN TO HIS      01301  0237  SATAN
DESERV'D NO SUCH RETURN FROM ME, WHOM HE      00464  0086  SATAN
CREATED WHAT I WAS IN THAT BRIGHT EMINENCE,   00505  0094  SATAN
; TO THEE I CALL, BUT WITH NO FRIENDLY VOICE,  00157  0029  SATAN
AGAINST HEAV'N'S MATCHLESS KING: AH           00410  0079  SATAN
MATCHLESS KING: AH WHEREFORE! HE DESERV'D NO  00425  0081  SATAN
FREE LOVE DEALT EQUALLY TO ALL? BE THEN HIS   01570  0297  SATAN
THEIR DIMINISHT HEADS; TO THEE I CALL, BUT    00136  0026  SATAN
I 'SDAIN'D SUBJECTION, AND THOUGHT ONE STEP   00783  0147  SATAN
WOULD SET ME HIGHEST, AND IN A MOMENT QUIT    00832  0155  SATAN
, AND IN A MOMENT QUIT THE DEBT IMMENSE OF    00852  0159  SATAN
PRAISE, THE EASIEST RECOMPENSE, AND PAY HIM   00646  0121  SATAN
THANKS, HOW DUE! YET ALL HIS GOOD PROV'D ILL  00690  0128  SATAN
I STILL RECEIV'D, AND UNDERSTOOD NOT THAT A   00984  0181  SATAN
GOOD UPBRAIDED NONE; NOR WAS HIS SERVICE      00561  0105  SATAN
SERVICE HARD. WHAT COULD BE LESS THAN TO      00594  0111  SATAN
GOOD PROV'D ILL IN ME, AND WROUGHT BUT        00719  0134  SATAN
POWERFUL DESTINY ORDAIN'D ME SOME INFERIOUR    01138  0209  SATAN
PART; BUT OTHER POWERS AS GREAT FELL NOT, BUT 01343  0245  SATAN
LOVE ACCURST, SINCE LOVE OR HATE, TO ME       01619  0296  SATAN
HAD RAIS'D AMBITION. YET WHY NOT? SOME OTHER   01230  0224  SATAN
I HATE THY BEAMS THAT BRING TO MY REMEMBRANCE  00245  0048  SATAN
GLORIOUS ONCE ABOVE THY SPHERE; TILL PRIDE     00319  0063  SATAN
DUE! YET ALL HIS GOOD PROV'D ILL IN ME, AND    00702  0130  SATAN

CALL, BUT WITH NO FRIENDLY VOICE, AND ADD THY 00172  0032  SATAN
HATE THY BEAMS THAT BRING TO MY REMEMBRANCE    00249  0049  SATAN
FELL, HOW GLORIOUS ONCE ABOVE THY SPHERE;      00311  0061  SATAN
WAS IN THAT BRIGHT EMINENCE, AND WITH HIS      00521  0097  SATAN
'SDAIN'D SUBJECTION, AND THOUGHT ONE STEP      00788  0148  SATAN
STILL RECEIV'D, AND UNDERSTOOD NOT THAT A      00988  0182  SATAN
HAD STOOD THEN HAPPY; NO UNBOUNDED HOPE HAD    01193  0217  SATAN
IN ME, AND WROUGHT BUT MALICE; LIFTED UP SO    00735  0137  SATAN
OF ENDLESS GRATITUDE, SO FORTHCOME, STILL      00831  0157  SATAN
HEAV'N AGAINST HEAV'N'S MATCHLESS KING: AH     00407  0078  SATAN
THAT BRIGHT EMINENCE, AND WITH HIS GOOD        00527  0099  SATAN
ME, AND WROUGHT BUT MALICE; LIFTED UP SO HIGH  00739  0138  SATAN
A MOMENT QUIT THE DEBT IMMENSE OF ENDLESS      00865  0161  SATAN
AND DISCHARG'D; WHAT BURDEN THEN? O HAD HIS    01089  0201  SATAN
ANGEL, I HAD STOOD THEN HAPPY; NO UNBOUNDED    01176  0215  SATAN
DEALS ETERNAL WOE.                             01670  0306  SATAN

: AH WHEREFORE! HE DESERV'D NO SUCH RETURN     00444  0083  SATAN

THAN TO AFFORD HIM PRAISE, THE EASIEST         00626  0117  SATAN
UP SO HIGH I 'SDAIN'D SUBJECTION, AND THOUGHT  00770  0144  SATAN
THE GOD OF THIS NEW WORLD; AT WHOSE SIGHT ALL  00075  0013  SATAN
FROM THY SOLE DOMINION LIKE THE GOD OF THIS    00044  0008  SATAN
DEALT EQUALLY TO ALL? BE THEN HIS LOVE         01584  0269  SATAN
ACCURST, SINCE LOVE OR HATE, TO ME ALIKE, IT   01623  0297  SATAN
OR HATE, TO ME ALIKE, IT DEALS ETERNAL WOE.    01643  0300  SATAN

; LIFTED UP SO HIGH I 'SDAIN'D SUBJECTION, AND 00762  0143  SATAN
KING: AH WHEREFORE! HE DESERV'D NO SUCH        00434  0082  SATAN
```